# Universidad de las Ciencias Informáticas


Facultad 15

# Herramienta para evaluar la portabilidad de los datos en el desarrollo de software.

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

**Autores:** Alicia del Carmen Pérez Roldán
Jorge Luis Fuoman Noguera

**Tutores: Ing.** Pedro Enrique Castiñeiras Sanchez
**Ing.** Julio Cesar Prieto Alvarez

Ciudad de la Habana

Junio del 2010

*El hombre nada puede aprender sino en virtud de lo que sabe.*

*Aristóteles*

**DECLARACIÓN DE AUTORÍA**

Declaramos ser los autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

**Alicia del Carmen Pérez Roldán**

_____

Firma del autor

**Jorge Luis Fuoman Noguera**

_____

Firma del autor

**Pedro Enrique Castiñeiras Sanchez**

_____

Firma del tutor

**Julio Cesar Prieto Alvarez**

_____

Firma del tutor

**DATOS DE CONTACTO**

Nombre: Pedro Enrique Castiñeiras Sanchez
Descripción: Ingeniero en Ciencias Informáticas, Adiestrado.
Correo electrónico: pesanchez@uci.cu

Nombre: Julio Cesar Prieto Alvarez
Descripción: Ingeniero en Ciencias Informáticas, instructor recién graduado.
Correo electrónico: jcprieto@uci.cu

## Agradecimientos

A mis padres por formarme como una mujer de bien, apoyarme siempre en todo y confiar en mí. ¡ Los adoro ¡

De manera especial a mi amor Julio Cesar, mi chiquito, que ha estado conmigo estos 5 años acompañándome en las buenas y en las malas, dándome fuerzas para seguir adelante en los momentos más duros y aguantando todas mis malcriadeces, pero por sobre todas las cosas dándome ese amor tan lindo. ¡Te quiero mi vida ¡

A mi hermano que lo quiero muchísimo.

A mi tía Estercita, mis primas Yuni, Imara y Yoisandra.

A mi cosita bella, mi ahijada María Karla.

A mi abuela Olga y mi abuelo Urra.

A mi tía Indira.

A mis amigas Clara, Meme, Ary, Yasmin y Nori por estar siempre ahí cuando las necesito.

A mi suegra por permitirme entrar en su casa y apoyarme siempre, por el cariño que me ha brindado.

A mi compañero de tesis Fuomi, por todo el apoyo durante la realización de este trabajo.

A mis tutores por toda la colaboración durante este tiempo.

A todas aquellas personas que de una forma u otra estuvieron presentes en algún momento durante estos 5 años, mis amigos, mis compañeros de grupo, mis profes y todos los que siempre o en algún momento estuvieron presente.

**A todos muchas Gracias**

*Alicia del Carmen Pérez Roldán*

## Agradecimientos

A mis padres Jorge y Olivia, por todo el amor, la dedicación y la confianza que me han brindado en todos los momentos de mi vida. ¡Los quiero mucho!

A mi abuela Cola por su cariño y por pedirle siempre a Dios por mí.

A mi hermano que seguro hubiera estado a mi lado ayudándome y guiándome.

A mis tíos Roque, Efrén, Crucito, Reyma, Isabel y Frank por toda la preocupación y el apoyo que me brindaron.

A mis primas Yamila, Reymita, Yaisel por quererme tanto, a Thalía, Nena, Arbelito por ser ustedes un ejemplo para mí, a Yosvani por todos sus consejos.

A mi hermano Juan, y a mis sobrinas, por todo el cariño que me tienen.

A mi novia Daylé por su comprensión y apoyo y a su familia que me acogieron como uno más entre ellos.

Al resto de mi familia que siempre me ayudaron y me dieron su apoyo.

A mis grandes amigos Adalberto, Mariño, Rodney, Alfredo Juan Carlos, Sariol, Mario Damián, Manuel, Barlía, Mario e Isabel, que me ayudaron y compartieron conmigo en los buenos y malos momentos.

A mi amiga Yami por ser tan especial.

A mi compañera de tesis Alicia por su empeño y dedicación y a su novio Julio por todo el apoyo y ayuda que nos dio.

A mi tutor por su apoyo y ayuda.

A todos mis compañeros, amigos, personas, que conocí en la UCI.

A la Revolución que me posibilitó llegar a donde he llegado.

**A todos muchas Gracias**

*Jorge Luis Fuoman Noguera.*

## Dedicatoria

A mi mamá y mi papá que han sido mi guía a seguir, mi mejor ejemplo y mis mejores amigos, a ellos que lo sacrifican todo por mí y que me guían siempre por el mejor camino, que están siempre ahí cuando los necesito, que me criaron y me dieron todo para ser la persona que soy hoy y a los que amo por encima de cualquier cosa.

A mi hermano para que el día de mañana siga el ejemplo de nuestros padres y se convierta en un profesional y un hombre de bien.

Muy especial a una persona que hoy no está a mi lado, pero que sé que estaría bien feliz de ver cómo me convierto en una profesional, a esa persona que me consintió y me complació en cuanto se me antojaba, que me dio un amor infinito y a quien voy a querer siempre aunque ya no esté. Para ti mami.

*Alicia del Carmen Pérez Roldán*

A mis padres por confiar en mí y por indicarme siempre el camino correcto a seguir. A mi hermano que ya no está y que nunca conoció el mundo le dedico este trabajo. A mi abuela, muy en especial por su cariño. A mi familia y amigos por todo el apoyo que me dieron. Y a todos aquellos que de una forma u otra hicieron posible este momento y que me han brindado su ayuda.

*Jorge Luis Fuoman Noguera.*

## Resumen

Debido a que en la Universidad de las Ciencias Informáticas se necesita hacer mejoras en los indicadores de calidad, como es el caso de la portabilidad de los datos que se manejan en los diferentes proyectos que en ella se desarrollan, se ha propuesto elaborar una herramienta que permita evaluar dicha portabilidad.

Debido a que hoy en día no existen métricas que permitan evaluar la portabilidad de los datos se realizó un estudio sobre diferentes características de los Sistemas Gestores de Bases de Datos PostgreSQL y Oracle dónde se definieron un conjunto de métricas con las cuales se elaboró la herramienta que va a permitir evaluar la portabilidad de los datos en el desarrollo de software.

Esta herramienta además de dar como resultado un porciento de afectación de la base de datos que se analice, ayuda a determinar dónde se encuentran los mayores problemas de portabilidad de la misma, siendo esto de gran beneficio para el usuario, puesto que tendría una idea de cuan portables son los datos que manejan para el caso de una posible migración.

**PALABRAS CLAVES**

Portabilidad, Bases de Datos, métricas, estándar.

Índice

## Introducción

Los avances tecnológicos producidos en los últimos años han generado un cambio en la forma de ver los sistemas de información y en general las aplicaciones computacionales. Actualmente, para las aplicaciones, existe una mayor demanda por mayor funcionalidad, mayor número de servicios, más flexibilidad, mejor rendimiento y gestión de un mayor flujo de información, así como una mejor adaptabilidad a diferentes entornos.

El almacenamiento de la información es un elemento importante a tener en cuenta a la hora de desarrollar una aplicación. Existen diversas formas de almacenar los datos, entre las que se encuentran los ficheros y la utilización de Bases de Datos principalmente.

El almacenamiento de información a través de ficheros se utiliza fundamentalmente en aplicaciones de escaso almacenamiento de información y donde esta no sea redundante. Esta vía es de gran ayuda a la hora de estructurar de forma adecuada toda la información, además permiten acceder a los datos de forma rápida.

El uso de sistemas de información por parte de las organizaciones requiere el almacenamiento de grandes cantidades de datos. Estos datos redundantes por demás, se emplean para generar resultados o para compartirlos con otros sistemas para lo cual la gestión de información a través de ficheros no es del todo una alternativa conveniente.

En la mayoría de estas aplicaciones donde el flujo de datos es grande y complejo se utilizan Sistemas Gestores de Bases de Datos (SGBD) para el almacenamiento y procesamiento de la información. Estos Gestores de Bases de Datos estructuran los ficheros de una mejor forma.

El uso de Gestores de Bases de Datos permite a los diferentes usuarios considerar la información como un recurso corporativo que carece de dueños específicos, elimina la información redundante o duplicada, permite compartir información de forma simultánea, ofrecen gran velocidad para la manipulación de la información, entre otras ventajas.

Existen diversos Gestores de Bases de Datos cada uno de ellos con sus características, ventajas y desventajas. En la Universidad de las Ciencias Informáticas (UCI) que es una institución que se dedica al desarrollo de sistemas de información tanto nacionales como internacionales, se utilizan estos gestores.

Los procesos de toma de decisiones de acuerdo a las funcionalidades de cada uno de los sistemas que se producen en la UCI cambian constantemente. Estos cambios se pueden producir por no haber desarrollado una correcta comprensión de los requisitos del sistema, que no haya un entendimiento común entre el equipo de desarrollo y el cliente o cambios inesperados en la configuración del sistema.

De acuerdo a la necesidad del cambio de las funcionalidades del sistema relacionadas con las versiones del producto a entregar, se hace necesario migrar de un gestor a otro o hacia otra versión del gestor que se esté utilizando. Generalmente los cambios que ocurren de una versión a otra de un software provocan pérdida de tiempo en la migración de la aplicación, la cual incluye además la migración de los datos.

Uno de los elementos claves a tener en cuenta es la portabilidad, para garantizar así una migración exitosa de los mismos.

La UCI como empresa productora de software necesita un sistema de indicadores que midan la calidad de la portabilidad de los datos.

Según la investigación realizada al campo de acción del presente trabajo tomándose una muestra de 23 proyectos con carácter heterogéneo se puede concluir:

- De los 23 proyectos en ninguno se tiene en cuenta la portabilidad de los datos que se manejan en los mismos.
- Solo el 30% ha estudiado la posibilidad de migrar de un gestor a otro o a otra versión del que se esté utilizando y las consecuencias que esto trae.
- De los 23 proyectos, en los 7 que utilizan Oracle una migración hoy en día no es posible debido a que en los mismos se explotan al máximo las ventajas que este gestor proporciona.

Tomando en cuenta la situación actual surge el siguiente **problema**: ¿Cómo evaluar la portabilidad de los datos para contribuir a la migración de los mismos en las soluciones informáticas desarrolladas en la UCI? Con el objetivo de dar solución al problema planteado se determinó como **objeto de estudio** el proceso de desarrollo de software, teniéndose como **objetivo** crear una herramienta que permita evaluar la portabilidad de los datos para la migración de los mismos en las soluciones informáticas desarrolladas en la UCI y delimitándose como **campo de acción** la portabilidad en la gestión de los datos en el desarrollo de las soluciones informáticas.

Para guiar la investigación se plantea la **siguiente idea a defender**:

Si se cuenta con una herramienta que evalúe la portabilidad de los datos entonces será posible medir la posibilidad de migrar la información que se maneja a otros gestores de bases de datos u otras versiones disponibles del mismo. Planteándose como **Objetivos específicos:**

- Realizar estudio del arte de las herramientas existentes para evaluar la portabilidad de los datos.
- Realizar un estudio de los principales aspectos teóricos referentes a la investigación entre los que se encuentran calidad de software, métricas de calidad de software y portabilidad de software.
- Establecer métricas para evaluar la portabilidad de los datos, a partir de las principales características de cada gestor.
- Elaborar la herramienta de evaluación.
- Realizar evaluación de la efectividad de la herramienta.

Los **Métodos Teóricos de investigación científica** utilizados son los siguientes:

*Hipotético Deductivo*: Este método sirvió para llegar a una solución para el problema planteado partiendo de la hipótesis trazada.

*Histórico Lógico:* Fue utilizado para conocer el estado actual del problema planteado y hacer un estudio de la evolución y desarrollo del objeto de estudio de la investigación.

*Analítico Sintético*: Se hizo un análisis del objeto de estudio en función de la problemática existente y se realizó una síntesis de la propuesta de las métricas a utilizar para darle solución al objetivo.

De los **Métodos Empíricos** se utilizaron el método de la *Observación Participante* donde el tiempo de trabajo en proyectos de software permitió identificar la problemática y la *Encuesta* donde a través de un conjunto de preguntas nos permitió obtener la percepción del encuestado acerca del fenómeno de investigación.

Con el desarrollo de este trabajo y como aporte práctico del mismo se dispondrá de una herramienta que permita evaluar la portabilidad de los datos.

Para dar cumplimiento a todas las tareas y objetivos trazados el presente trabajo está dividido en tres capítulos.

**Capítulo 1:** En este capítulo se realiza un estudio del arte vinculado al campo de acción y a la propuesta de este trabajo. Se describen detalladamente los conceptos principales a utilizar durante el desarrollo de la investigación para lograr un mejor entendimiento de la misma.

**Capítulo 2:** En este capítulo se obtienen las métricas necesarias para evaluar la portabilidad de los datos teniendo en cuenta las principales características de cada Gestor.

**Capítulo 3:** Se hace una descripción y evaluación de la herramienta propuesta.

# 1. Fundamentación teórica.

El objetivo fundamental de este capítulo es exponer los fundamentos generales que sustentan teóricamente la solución del problema.

Se abordan aspectos sobre las principales tendencias en el manejo de los datos a nivel nacional e internacional, así como de los conceptos de calidad, portabilidad, lenguajes de consulta de datos. Además se realiza un estudio de las principales características de diferentes Gestores de Bases de Datos.

## 1.1. Estado del arte.

La portabilidad de los datos es un factor importante a tener en cuenta en el momento de medir la calidad de un sistema informático. Queda en manos del equipo de desarrollo, arquitecto fundamentalmente, determinar cuan portable serán los datos que gestiona el sistema que se pretende desarrollar. Pero surge la interrogante de ¿Cómo medir si la información que maneja el sistema es realmente portable o no?

A nivel internacional el proceso de evaluación es realmente complejo puesto que existen muchos aspectos a tener en cuenta y que muchas veces no se les presta atención o el tiempo que se destina para estas investigaciones es muy pequeño. Actualmente escasean las herramientas que permiten evaluar la calidad de software y por ende la portabilidad de los datos.

En investigaciones realizadas en la UCI, se ha podido determinar, que en esta no se tiene en cuenta la portabilidad de los datos que se manejan en los sistemas que se desarrollan en la misma, figurando este aspecto dentro de los requerimientos no funcionales del sistema.

La encuesta realizada en 23 de los proyectos que se desarrollan en la universidad arrojó como resultado que en 7 (30 %) de los proyectos se tiene en cuenta la posibilidad de una futura migración, ya sea entre SGBD o entre versiones de un SGBD pero no tienen una vía específica para realizar esta operación, y el 100% de ellos no conoce ningún mecanismo que les permita evaluar la portabilidad de sus datos.

Por otro lado no existe una herramienta que permita medir la portabilidad de dichos datos.

## 1.2. Calidad

“ La calidad de un producto es la capacidad de lograr los objetivos por el cual fue construido, es la percepción que el cliente tiene del mismo, es una fijación mental del consumidor que asume conformidad con dicho producto o servicio ” (1).

Por consiguiente, el propósito de la calidad es proporcionarle al cliente una oferta apropiada con sus necesidades y hacer que se cumplan con las especificaciones requeridas de lo que se quiere hacer. La norma ISO 9000 define la calidad como el “grado en el que un conjunto de características inherentes cumple con los requisitos” por su parte, el Diccionario de la Real Academia Española, conceptúa la Calidad como la “propiedad o conjunto de propiedades inherentes a algo, que permiten juzgar su valor", y es sinónimo de "buena calidad" la "superioridad o excelencia". (2)

### Estándar de Calidad

Un estándar define el rango en el que resulta aceptable el nivel de calidad que se alcanza en un determinado proceso para un indicador, si el valor del indicador está dentro del rango significa que se está cumpliendo con el criterio de calidad definido. Si, por el contrario, está por debajo del rango significa que no se están cumpliendo los compromisos de calidad y si está por encima se está gastando en términos de esfuerzo más de lo planeado. Por lo tanto un estándar de calidad óptimo es el que reúne los requisitos mínimos en busca de la excelencia dentro de una organización institucional. (3)

### Modelo de calidad

Un Modelo de Calidad no es más que un conjunto de buenas prácticas para el ciclo de vida del software, enfocado en los procesos de gestión y el desarrollo de proyectos. Te dicen que hacer, no como hacerlo, porque dependen de las metodologías que se utilicen y de los objetivos del negocio en cuestión, o sea es una guía donde se recogen un grupo de parámetros que hacen que un producto tenga calidad. (4)

**Norma de calidad**

Es un documento aprobado por un organismo reconocido, que proporciona reglas, pautas y/o características para uso común. Es un conjunto de acuerdos documentados que contienen especificaciones técnicas u otros criterios precisos para ser usados constantemente, como reglas, lineamientos o definiciones de características con el fin de conseguir un grado óptimo de orden en el contexto de la calidad. (5)

Algunas de las normas de calidad existentes se mencionan a continuación:
ISO 9001: Aplicable cuándo la conformidad con los requisitos especificados debe ser asegurada por el suministrador durante varias etapas que pueden incluir el diseño/desarrollo, desarrollo, la producción, la instalación y el servicio post venta.
ISO 9002: Aplicable cuándo la conformidad con los requisitos especificados debe ser asegurada por el suministrador durante la producción y la instalación.
ISO 9003: Aplicable cuándo la conformidad con los requisitos especificados debe ser asegurada por el suministrador únicamente en la inspección y ensayos finales.
ISO 9004: Recoge las directrices para la gestión de la calidad, aplicable a todas las organizaciones.

**Calidad de Software.**

Existen diversas definiciones sobre qué es la calidad de software, Pressman en 1992 definió la calidad de software como: *"*Concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos con los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente" (6)
El concepto a utilizar es el que mide la capacidad que tiene la solución informática para satisfacer las necesidades del cliente, el cual está compuesto por un conjunto de indicadores tales como: flexibilidad, corrección, confiabilidad, mantenibilidad, *portabilidad*, usabilidad, seguridad e integridad.

## 1.3. Métricas

Las métricas describen muchos casos de medición, según el Dr. Lionel C. Briand's son medidas estadísticas útiles para entender, monitorizar, controlar, predecir y probar el desarrollo de determinado proceso. El Dr. Shari Lawrence Pfleeger define que las métricas pueden ser utilizadas para que los profesionales e investigadores puedan tomar las mejores decisiones.

Las métricas ayudan a entender tanto el proceso técnico que se utiliza para desarrollar un producto, como el propio producto. En el proceso para intentar mejorar un producto, el mismo se mide para aumentar su calidad. En general, la medición persigue tres objetivos fundamentales: ayudar a entender qué ocurre durante el desarrollo y el mantenimiento, permitir controlar qué es lo que ocurre en los proyectos y poder mejorar los procesos y los productos.

Según Pressman una métrica tiene como características deseables las siguientes:

1. Simple y fácil de calcular: Debe ser de ayuda al ingeniero y si es posible, deberá ser diseñada para poderla computar automáticamente.
2. Empírica y persuasiva: Debe obtenerse de la práctica y disminuciones o incrementos de sus valores indicar algún síntoma.
3. Consistente: Debe ser objetivamente calculada con la menor ambigüedad posible, no importa la persona que la tome.

**Métricas de calidad de software**

Las métricas se pueden definir como:

“La continua aplicación de técnicas basadas en la medición al proceso de desarrollo de software y a sus productos para proveer información administrativa, significativa y oportuna, junto con el uso de esas técnicas para mejorar el proceso y sus productos”. (7)

El IEEE “Standard Glosary of Software Engering Terms” define métrica como “una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado”. (8)

“Las métricas son un buen medio para entender, monitorizar, controlar, predecir y probar el desarrollo software y los proyectos de mantenimiento”. (9)

“Es una medida estadística no cuantitativa que se aplica a todos los aspectos de calidad de software, los cuales deben ser medidos desde diferentes puntos de vista durante todo el desarrollo del producto”. (6)

Existen diversos tipos de métricas entre las cuales se encuentran las métricas de producto, las métricas de procesos, las métricas basadas en atributos internos del producto y las métricas basadas en atributos externos del producto dentro de esta última se encuentran las métricas de portabilidad, clasificación dentro de la que se encuentran las métricas a definir en la presente investigación. Sobre las métricas de portabilidad actualmente existe muy poca investigación.

Para esta investigación las métricas proporcionarán una medida de cuan portables son los datos que se manejan en determinado software.

## 1.4. Portabilidad de software.

La portabilidad, en informática, se refiere a la capacidad de un programa o sistema de ejecutarse en diferentes plataformas o arquitecturas con mínimas modificaciones. El código fuente del software es capaz de reutilizarse en otros ambientes, en vez de tener que modificar este cuando el software pasa de una plataforma a otra. A mayor portabilidad menor es la dependencia del software con respecto a la plataforma. (10)

Esta definición de portabilidad de software se ajusta al objetivo de esta investigación la cual tiene como propósito evaluar la portabilidad de los datos que se manejan en un determinado sistema, lo cual contibuye en general a la portabilidad del software.

El concepto a utilizar en la presente investigación es el que está dado por la capacidad que tiene un conjunto de datos, que se manejen en determinado software, de comportarse de igual forma en diferentes SGBD o en diferentes versiones del SGBD que se esté utilizando.

## 1.5. Lenguajes de consulta de datos

Un lenguaje de consultas es un lenguaje informático usado para hacer consultas a bases de datos y sistemas de información. Es el lenguaje a través del cual se solicita información de la base de datos.

Structured Query Language (SQL, por sus siglas en inglés) empieza en 1970, con el nombre de SEQUEL el cual terminaría siendo el predecesor de SQL que pasaría a ser el lenguaje por excelencia de los diversos SGBD relacionales siendo estandarizado por fin en 1986 por el ANSI, con la definición, por parte de IBM, de un "lenguaje para la especificación de las características de las bases de datos que adoptaban el modelo relacional". SQL permite crear, actualizar y manipular las bases de datos. Está compuesto por cláusulas, funciones, operadores y comandos los cuales se clasifican en cuatro tipos:

- Lenguaje de definición de datos (DDL, por sus siglas en inglés): Las sentencias DDL son utilizadas para la creación de una base de datos y todos sus componentes: tablas, índices, relaciones, disparadores (más conocidos por triggers), procedimientos almacenados, etc.
- Lenguaje de manipulación de datos (DML por sus siglas en inglés): Las sentencias DML, son aquellas que se utilizan para insertar, borrar, modificar y consultar los datos de una base de datos.
- Lenguaje de control de datos (DCL por sus siglas en inglés): Estos comandos se utilizan para controlar el acceso a los datos en la base de datos ya sea por la concesión del permiso de acceso o por la revocación de la autorización ya concedida. Los comandos SQL que pertenecen a esta categoría son:
    - GRANT: utilizado para la concesión del permiso para acceder a los datos.
    - REVOKE: utilizado para revocar los permisos ya concedidos para acceder a los datos.
- Lenguaje de control de transacciones (TCL por sus siglas en inglés): Estos comandos se utilizan para controlar el estado de la transacción que se disparan. El control de una transacción incluye capacidades de cometerlo, revertir los cambios, identificar los puntos de retorno para

controlar el ROLLBACK, etc. Los comandos que pertenecen a esta categoría son los siguientes:

- COMMIT: utilizado para hacer los cambios permanentes.
- ROLLBACK: utilizado para la restauración de la base de datos para el Estado cometidos anteriormente.
- SAVEPOINT: utilizado para la identificación de un punto en que la transacción se puede revertir si se requiere, por supuesto, antes de que se confirmó la transacción.
- SET TRANSACTION: se utiliza para modificar las propiedades de la transacción actual, como modificar el nivel de aislamiento de la transacción.

Gracias al éxito que tuvo el empleo de SQL internamente en IBM otras compañías empezaron a desarrollar sus productos relacionales basados en SQL convirtiéndose en el estándar industrial en lo que respecta a las bases de datos relacionales.

Por esta razón en 1986, el ANSI adoptó SQL como estándar para los lenguajes relacionales y en 1987 se transformó en estándar ISO. Esta versión del estándar va con el nombre de **SQL/86** y la ISO correspondiente a la misma es la ISO 9075: 1987, esta fue diseñada para manipular y para recuperar los datos almacenados en el producto original de la base de datos. Sin embargo, este primer estándar no cubre todas las necesidades de los desarrolladores e incluye funcionalidades de definición de almacenamiento que se consideraron suprimir por lo que este ha sufrido diversas revisiones que han conducido a diferentes versiones:

- **SQL/89:** Le corresponde la ISO/IEC 9075: 1989 basada en la anterior pero con una serie de mejoras (definición de claves primarias, integridad de los datos, etc.). Define una característica nueva e importante respecto a la versión anterior que es la posibilidad de utilizarse a través de dos interfaces: interactivamente o dentro de programas de aplicación.
- **SQL/92:** ISO / IEC 9075:1992, constituye una revisión de SQL/89 la cual proporciona nuevas palabras reservadas, reglas de sintaxis y caracteres comodines que mejoran la posibilidad de crear consultas, filtros e

instrucciones SQL. Una característica nueva que incorpora es el uso de constructores de filas para insertar múltiples filas a la vez, con una sola sentencia SQL.

- **SQL/1999:** La versión anterior sufre 5 revisiones las cuales quedan recogidas en las ISO/IEC 9075-1:1999, ISO/IEC 9075-2:1999, ISO/IEC 9075-3:1999, ISO/IEC 9075-4:1999 e ISO/IEC 9075-5:1999. Estos estándares de forma general incorporan nuevas características entre las que se destacan la definición de las estructuras de datos y las operaciones de base de datos en SQL, proporcionar capacidades funcionales para la creación, acceso, mantenimiento, control y protección de datos de SQL así como especificaciones de la sintaxis y la semántica de las declaraciones para añadir una capacidad de procedimiento para el lenguaje SQL en las funciones y procedimientos. Este incluye declaraciones para dirigir el flujo de control, definir variables, hacer tareas y manejar las condiciones de excepción.
- **SQL/2003:** Esta versión presenta 9 revisiones (ISO/IEC 9075-1:2003, ISO/IEC 9075-2:2003, ISO/IEC 9075-3:2003, ISO/IEC 9075-4:2003, ISO/IEC 9075-9:2003, ISO/IEC 9075-10:2003, ISO/IEC 9075-11:2003, ISO/IEC 9075-13:2003, ISO/IEC 9075-14:2003). Estas de forma general incorporan definiciones a las extensiones de SQL para apoyar la gestión de datos externos, definen las extensiones al lenguaje SQL que admite la incrustación de SQL en programas escritos en el lenguaje de programación Java permitiendo invocar métodos estáticos escritos en este lenguaje de programación, además específicamente la ISO/IEC 9075-14:2003 define las formas en que el lenguaje SQL puede utilizarse en combinación con XML definiendo la forma de la importación y almacenamiento de datos XML en una base de datos SQL.
- **SQL/2006:** En esta versión se realiza una sola revisión específicamente a la ISO/IEC 9075-14:2003 recogida en el estándar ISO/IEC 9075-14:2006 que le incorpora la característica de proporcionar servicios que permiten a las aplicaciones para integrar en su código SQL el uso de XQuery.

- **SQL/2008:** Esta versión presenta 9 revisiones (ISO/IEC 9075-1:2008, ISO/IEC 9075-2:2008, ISO/IEC 9075-3:2008, ISO/IEC 9075-4:2008, ISO/IEC 9075-9:2008, ISO/IEC 9075-10:2008, ISO/IEC 9075-11:2008, ISO/IEC 9075-13:2008, ISO/IEC 9075-14:2008). Es compatible con cláusulas ORDER BY en las expresiones de consulta de alto nivel, sub-consultas y vistas. Esta funcionalidad es ya compatible con SQL Anywhere. Añade la sentencia TRUNCATE.

## 1.6. Sistemas Gestores de Bases de Datos (SGBD):

Un Sistema Gestor de base de datos (SGBD) es un tipo de software muy específico o conjuntos de ellos que permite manejar de manera clara, sencilla y ordenada altos volúmenes de datos. Actualmente existe una amplia gama de SGBD con características propias, no obstante todos deben tener en cuenta los siguientes aspectos de manera general: abstracción de la información, la independencia de los datos, redundancia mínima, consistencia en los datos, seguridad, integridad, respaldo y recuperación, tiempo de respuesta y control de concurrencia.

Existen SGBD muy potentes tanto en la clasificación de software propietario como software libre. Como propietarios podemos encontrar Oracle, Microsoft SQL Server y MySQL, este último muy utilizado en la Web por su simplicidad de uso, que por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Estos SGBD lideran el mercado por sus altas prestaciones dado muchos años de experiencia, mientras que como opción libre se encuentra, entre otros, PostgreSQL que si bien no soporta alguno de los aspectos más avanzados si representa una opción muy confiable y comprometedora. (12)

### 1.6.1. Microsoft SQL Server (MS SQL)

Producido por Microsoft. Su principal lenguaje de consulta es Transact SQL, una implementación del lenguaje de consulta estructurado ANSI/ISO usado

tanto por Microsoft como por Sybase. SQL Server es comúnmente usado por empresas que necesitan de pequeñas a medianas bases de datos, aunque en los últimos años ha sido ampliamente adoptado por empresas que poseen grandes bases de datos empresariales. Microsoft SQL Server constituyó la entrada de Microsoft hacia el mercado de bases de datos de nivel empresarial, compitiendo con Oracle, IBM, y luego, el propio Sybase. La versión SQL Server 2005, es un potente gestor de bases de datos, que incorpora numerosas características completamente novedosas, lo que lo hace un producto extremadamente costoso. (13)

### 1.6.2. Mysql

Es un sistema de gestión de bases de datos relacional, un software de código abierto que por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Utiliza SQL como lenguaje de programación, en sus inicios carecía de elementos esenciales en las bases de datos relacionales, pero poco a poco estos elementos están siendo incorporados tanto por desarrolladores internos como por desarrolladores de software libre. Es una herramienta que brinda velocidad al realizar las operaciones, lo que le hace uno de los gestores con mejor rendimiento. Tiene un bajo costo en requerimientos para la elaboración de bases de datos, ya que debido a su bajo consumo puede ser ejecutado en una máquina con escasos recursos sin ningún problema. Brinda facilidad de configuración e instalación, soporta gran variedad de sitemas operativos, posee baja probabilidad de corromper datos, incluso si los errores no se producen en el propio gestor, sino en el sistema en el que está, implementa funcionalidades Web que permiten un acceso a los datos, seguro y fácil desde Internet. Es uno de los SGBD más populares desarrollado bajo la filosofía de código abierto. Entre sus desventajas se puede decir que un gran porcentaje de sus utilidades no están documentadas y no es instuitivo como otros programas ejemplo Acces el cual es un programa de sistema de gestión de base de datos relacional. Es un

componente de la suite Microsoft Office que permite crear ficheros con bases de datos que pueden ser consultados por otros programas. (14)

### 1.6.3. Oracle

Entre los más usados dentro del grupo de los SGBD con licencias comerciales se encuentra Oracle, el cual es considerado el SGBD más completo que existe. Sus características más destacadas son el soporte de transacciones, gran estabilidad, seguridad, escalabilidad, así como que es un sistema multiplataforma, entre otras ventajas.

En sus inicios fue muy revolucionario dado que usaba la filosofía de bases de datos relacionales, algo que por los años setenta, fecha en la que surge Oracle, era todavía desconocido. Hasta hace poco su dominio en el mercado de los servidores de bases de datos empresariales era casi total, pero recientemente está sufriendo la competencia de Microsoft SQL Server y la oferta de los sistemas gestores de bases de datos libres. (12)

A pesar de considerarse como el SGBD más completo presenta inconvenientes tales como:

- Es un software propietario, además su elevado precio (el SGBD más caro) hace que solo se vea en empresas muy grandes y multinacionales, por norma general.
- Elevado costo de soporte técnico.
- Infinidad de cursos para poder manejar una sola línea (DBA, Desarrollo, Recursos Humanos, Minería de Datos, entre otras muchas).
- Múltiples vulnerabilidades han sido identificadas en este SGBD como ataques de inyección de SQL, evitar restricciones de seguridad, manipulación de datos, etc. que de ser explotadas por atacantes maliciosos podrían ocasionar importantes perjuicios. Diez parches de actualización fueron publicados en enero del 2010 para este gestor, con el objetivo de tapar los agujeros existentes en versiones actuales y pasadas de las bases de datos de Oracle, los cuales se catalogaron como de alto riesgo a partir del momento en que un

atacante podría explotar el acceso a un servidor sin necesidad de emplear una cuenta de usuario.

También presenta una larga lista de ventajas que lo dejan ver como el SGBD más completo existente en la actualidad.

- Es manejador de BD relacional que hace uso de los recursos del sistema informático en todas las arquitecturas de hardware, para garantizar su aprovechamiento al máximo en ambientes cargados de información.
- Oracle puede ser usado tanto para el manejo de información personal, como para gigantescas bibliotecas multimedia, y corre en laptops, computadoras personales (PC), microcomputadoras, mainframes y computadoras con procesamiento paralelo masivo.
- Soporta la mayoría de los lenguajes de computación al igual que 26 idiomas diferentes.
- Corre automáticamente en más de 80 arquitecturas de hardware y software distinto sin tener la necesidad de cambiar una sola línea de código, esto se debe a que más del 80% de los códigos internos de Oracle son iguales a los establecidos en todas las plataformas de sistemas.
- Soporta datos alfanuméricos ubicados en las tradicionales "filas y columnas" de las BD, y también soporta textos sin estructura, imágenes, audio y video.
- Incluye mejoras de rendimiento y de utilización de recursos.
- Soporta aplicaciones de procesamiento de transacciones online (OLTP) y de data warehousing mayores y más exigentes.
- Está disponible en múltiples plataformas como Windows, Linux, todas las versiones de Unix ofrecidas por diversas empresas como IBM, Sun, Digital, HP, Sequent, etc. y también en VAX-VMS, así como en MVS. La naturaleza multiplataforma de Oracle, lo convierte en una verdadera solución empresarial.

- Bloqueo y concurrencia: El lector verá los datos tal y como estaban antes de que el que escribe comenzará a cambiarlos ( hasta que este haga commit).
- Disponibilidad. Soporta ambientes de cluster en modo activo-pasivo, es decir que un solo nodo utiliza la base de datos mientras el/los otro/s nodo/s está/n pendientes de entrar en funcionamiento en el momento que el servidor primario tenga una falla. Permite tener copias de la base de datos productiva en lugares lejanos a la ubicación principal, esta funcionalidad se le denomina Oracle Standby Database. Las copias de la Base de Datos productiva pueden estar en modo de lectura-solamente.

Por todas las características, ventajas y desventajas de este gestor y por los resultados obtenidos en la encuesta, se determinó que Oracle constituye uno de los gestores más utilizados en la UCI, fundamentalmente para el desarrollo de software con carácter internacional.

### 1.6.4. PostgreSQL

PostgreSQL está considerado como el sistema gestor de bases de datos de código abierto más avanzado del mundo. Proporciona un gran número de características que normalmente sólo se encontraban en las bases de datos comerciales de alto calibre tales como Oracle. Además de ser un sistema gestor de bases de datos relacional. (15)

Presenta un gran número de características como:

- DBMS Objeto-Relacional: PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multi-versión, soporte multi-usuario, transacciones, optimización de consultas, herencia, y arreglos.
- Altamente extensible: PostgreSQL soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario.

- Soporte SQL Comprensivo: PostgreSQL soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins) SQL92.
- Integridad referencial: PostgreSQL soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.
- API Flexible: La flexibilidad del API de PostgreSQL ha permitido a los vendedores proporcionar soporte al desarrollo fácilmente para el RDBMS PostgreSQL. Estas interfaces incluyen Object Pascal, Python, Perl, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C++, y Pike.
- Lenguajes procedurales: PostgreSQL tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL. Otra ventaja de PostgreSQL es su habilidad para usar Perl, Python, o TCL como lenguaje procedural embebido.
- Control de Concurrencia Multi-Versión*. (MVCC):* Es la tecnología que PostgreSQL usa para mantener copias sobre los datos de forma paralela, para acelerar el sistema de escritura de datos, haciendo un control de concurrencia entre las distintas versiones que se van escribiendo, para así proporcionar el acceso concurrente a la base de datos evitando el problema de que procesos lectores estén esperando a que se termine de escribir.
- Write Ahead Logging (WAL): Esta característica incrementa la dependencia de la BD al registro de cambios antes de que estos sean escritos en la base de datos. Esto garantiza que en caso de que la BD se caiga, existirá un registro de las transacciones a partir del cual podremos restaurar la BD desde el punto en que se quedó.
- Es un gestor bajo licencia Berkeley Software Distribution (BSD), que posee una gran escalabilidad, haciéndolo idóneo para su uso en sitios Web que posean alrededor de 500.000 peticiones por día. Además por su arquitectura de diseño, escala muy bien al aumentar el número de CPUs (Central Processing Unit) y la cantidad de memoria RAM (Random Access Memory en español memoria de acceso aleatorio).

- Soporta los tipos de datos básicos, así como: tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP...), cadenas de bits, etc.

Por todas las características, ventajas y desventajas de este gestor y por los resultados obtenidos en la encuesta se determinó que PostgreSQL constituye uno de los gestores más utilizados en la UCI.

## 1.7. Conclusiones

En el presente capítulo se trataron diferentes conceptos que son de vital importancia para dar solución al problema planteado. El concepto más importante tenido en cuenta es el de portabilidad de software, debido a que constituye el centro de la investigación.

Atendiendo a la problemática existente y a partir de la necesidad de elaborar una herramienta que permita evaluar la portabilidad de los datos, se llegó a las siguientes conclusiones:

Debido a las características, ventajas y desventajas de cada uno de los gestores y complementado con los resultados de la encuesta se llegó a la conclusión de que los gestores más utilizados y en los cuales va a estar centrada la investigación son PostgreSQL y Oracle.

Se decidió realizar un análisis exhaustivo de las principales características de ambos gestores con el objetivo de establecer las métricas necesarias que permitan evaluar la portabilidad entre los mismos.

# 2. Desarrollo de las métricas para evaluar la portabilidad.

En este capítulo se hace un análisis de las diferencias que existen entre los SGBD PostgreSQL y Oracle y entre las versiones de PostgreSQL, para analizar el problema planteado anteriormente de deficiencias en la portabilidad de los datos en el desarrollo de soluciones informáticas.

El capítulo tiene como objetivo:

- Definir las métricas necesarias para evaluar dicha portabilidad teniendo en cuenta las características más relevantes de estos SGBD.
- Definir un costo para cada una de aquellas características que afectan la portabilidad, teniendo en cuenta el esfuerzo que implicaría realizar un cambio de estas en el modelo de datos del proyecto para lograr la portabilidad. Este costo está dado en un rango de 0.1 a 1 de la siguiente forma:

  Costo bajo: 0.1 a 0.4

  Costo medio: 0.5 a 0.7

  Costo alto: 0.8 a 1

## 2.1. Portabilidad entre PostgreSQL y Oracle.

A continuación se relacionan un grupo de características que van a permitir establecer las métricas necesarias para evaluar las portabilidad entre los SGBD PostgreSQL y Oracle.

**Actualizaciones de Vistas**

Según lo estandarizado a nivel internacional las vistas son parte de la norma y pueden ser actualizadas siempre que tenga sentido. SQL 2008 tiene un complicado conjunto de normas que regulan cuando una vista es actualizable, básicamente diciendo que la vista es actualizable siempre y cuando la actualización se traduzca en un cambio sin ambigüedades. SQL 92 es más estricta, especificando que vistas actualizables no pueden derivarse de más de una tabla base. En PostgreSQL no se cumple con la

norma, pues no permite la actualización de vistas; ofrece un sistema no estándar de reglas como solución alternativa. Oracle, por su parte, cumple con lo estandarizado por SQL 92. De acuerdo con lo planteado se puede decir que en cuanto a este aspecto ambos SGBD no son compatibles.

En el caso de esta característica se le otorga un ***costo alto*** de valor ***1*** pues sería complejo poder realizar las actualizaciones de vistas de igual forma en ambos gestores, debido a que existen vías completamente diferentes para realizar esta operación, cumpliendo uno de ellos con lo estandarizado a nivel internacional mientras que el otro por su parte ofrece un conjunto de reglas no estándares.

**Joins**

El Join es utilizado en los SGBD para combinar registros de dos o más tablas. Todos los Sistemas Gestores de Base de Datos (DBMS por sus siglas en inglés) soportan el INNER JOIN básico, pero varios de ellos soportan otros tipos de join, como:

- Natural Left Join
- Using Join
- Full Join (puede ser la unión del Left Join y el Right Join)
- Cross Join

En cuanto a este aspecto se puede decir que ambos sistemas gestores de base de datos (SGBD) son totalmente compatibles pues ambos soportan los mismos tipos de Join.

En el caso de esta característica debido a que no existe diferencia alguna en la forma de realizar esta operación en ambos SGBD el ***costo*** sería ***0***.

**Copia de estructura**

El objetivo de la copia de estructura es que se pueda copiar una tabla t1 para una nueva tabla t2 sin necesidad de copiar los datos, solo la estructura

de la tabla. El estándar proporciona una característica opcional que define la cláusula LIKE para la definición de tabla usándose de la forma:

- *CREATE TABLA t2 (LIKE t1)*

Otra característica, que es una extensión de la anterior, permite que más características de una tabla puedan ser copiadas.

- *CREATE TABLE t2 (LIKE t1 INCLUDING IDENTITY INCLUDING DEFAULTS INCLUDING GENERATED)*

Si no se especifica cada una de las características entonces las mismas no serán parte de la nueva tabla. Los disparadores, las restricciones CHECK y otras características triviales no serán copiados a la nueva tabla. PostgreSQL cumple con el núcleo de la primera característica y solo admite parcialmente la segunda característica, además incluye algunas opciones no estándares como:

- Las opciones INCLUDING IDENTITY e INCLUDING GENERATED no son compatibles.
- Añade el INCLUDE CONSTRAINTS y el INCLUDE INDEXES.

No permite copiar la estructura de una vista usando *CREATE TABLE… LIKE…*, para realizar esta operación lo hace de la siguiente manera:

- *CREATE TABLE copytable AS SELECT * FROM viewname WHERE false.*

Oracle por su parte no es compatible con el estándar, este permite copiar la estructura de una tabla usando *CREATE TABLE… AS* combinándola con la cláusula WHERE quedando de la siguiente forma:

*CREATE TABLE t2 AS SELECT * FROM t1 WHERE 1<>1*

En cuanto a este aspecto ambos SGBD no son compatibles pues no tiene la misma forma de copiar una estructura, incluyéndole a esto que Oracle no cumple con nada de lo estandarizado para esta operación.

El ***costo*** en el caso de la copia de estructura tiene un valor de ***1*** pues en este caso se hace uso de una cláusula en uno de los gestores, simplificando el código para esta operación, mientras que por su parte el otro gestor no

admite el uso de esta cláusula haciendo más complejo el proceso de la copia de estructura.

**Organizar un conjunto de resultados**

La norma no especifica con que valores NULL debe ser ordenado en comparación con los valores no NULL, salvo que con cualquiera de los dos valores NULL se considera ordenado y que lo nulos se deben ordenar por encima o por debajo de todos los valores no NULL. Sin embargo el SGBD puede, opcionalmente, permitir al usuario especificar si se debe ordenar valores NULL primero o últimos de la siguiente manera:

... *ORDER BY... NULLS FIRST*

*o*

*... ORDER BY... NULLS LAST*

En PostgreSQL al igual que en Oracle de forma predeterminada los valores NULL se consideran superiores que cualquier valor no NULL, este comportamiento puede cambiar la clasificación añadiendo NULL FIRST o NULL LAST a la expresión ORDER BY característica que se añade en PostgreSQL a partir de la versión 8.3. Por lo que se puede decir que son compatibles en cuanto a este aspecto, pues ambos SGBD tienen la misma forma de realizar esta operación y a su vez ambos cumplen con lo estandarizado con la excepción de versiones anteriores a la 8.3 de PostgreSQL.

En el caso de esta característica solo para versiones anteriores a la 8.3 de PostgreSQL posee un ***alto costo*** de valor ***1*** pues para versiones superiores a esta el costo es 0 pues estas si son compatibles con Oracle.

**Límite simple**

El objetivo del límite simple es obtener una determinada cantidad de filas de una tabla. Según lo estandarizado se proporcionan 3 formas de realizar esta operación:

- Usando el *FETCH FIRST:*

*SELECT... FROM ... WHERE ... ORDER BY ... FETCH FIRST n ROWS ONLY*

- Usando la función Windows:

*SELECT * FROM (SELECT ROW_NUMBER () OVER (ORDER BY key ASC) AS rownumber, columns FROM tablename) AS foo*
*WHERE rownumber <= n*

- Usando cursor: Si la solicitud es con estado entonces se puede utilizar los cursores lo que implica:

*DECLARE cursor-name CURSOR FOR ...*
*OPEN cursor-name*
*FETCH ...*
*CLOSE cursor-name*

PostgreSQL soporta todos los estándares basados en el enfoque. En las antiguas versiones de este (PostgreSQL 8.3 y anteriores) se utilizó el siguiente método especial y específico de PostgreSQL:

*SELECT columns*
*FROM tablename*
*ORDER BY key ASC*
*LIMIT n*

Oracle, por su parte, soporta ROWS_NUMBER y no admite FETCH FIRST. Como Oracle no permite el AS en las subconsultas para nombrar, modifica ligeramente el código de SQL estándar quedando:

*SELECT * FROM (SELECT*
*ROW_NUMBER () OVER (ORDER BY key ASC) AS rownumber,*
*columns FROM tablename)*
*WHERE rownumber <= n*

En cuanto a esta característica ambos SGBD no son compatibles pues como se plantea anteriormente Oracle no admite el uso de FETCH FIRST mientras que PostgreSQL si lo admite.

A pesar de que Oracle no admite el uso del FETCH FIRST, esta característica tiene un ***costo medio*** de valor ***0.6*** pues existe otra forma de

realizar el límite simple según lo establecido en el estándar que es mediante la función Windows donde solo hay que realizar una ligera modificación que sería eliminar el AS utilizado para renombrar en la subconsulta para que sea común para ambos gestores.

**Insertar varias filas en un momento:**

Un uso práctico de los constructores de filas es el valor al insertar varias filas en un momento, lo cual según el estándar se realiza de la siguiente manera:

*INSERT INTO tablename VALUES (0,'foo') , (1,'bar') , (2,'baz');*

Que puede ser leído como una forma abreviada de:

*INSERT INTO tablename VALUES (0,'foo');*

*INSERT INTO tablename VALUES (1,'bar');*

*INSERT INTO tablename VALUES (2,'baz');*

PostgreSQL soporta el estándar desde la versión 8.2, mientras que Oracle lo hace de una forma específica:

```
INSERT INTO tablename
SELECT 0,'foo' FROM DUAL
UNION ALL
SELECT 1,'bar' FROM DUAL
UNION ALL
SELECT 2,'baz' FROM DUAL
```

Por lo que se puede decir que en cuanto a este aspecto ambos SGBD no son compatibles.

Esta característica posee un ***costo alto*** de valor ***1*** pues en Oracle solo a través del DUAL TABLE es posible insertar varias filas en un momento, no realizándose de igual forma para PostgreSQL.

**El tipo de dato BOOLEAN:**

Según lo estandarizado se establecen tres formas de ver al valor BOOLEANO:

1. VERDADERO

2. FALSO
3. DESCONOCIDO o NULL

PostgreSQL cumple con el estándar, reconoce al NULL como un literal BOOLEANO pero no acepta a DESCONOCIDO como literal BOOLEANO, mientras que Oracle por su parte no admite el valor BOOLEANO, en su lugar se puede trabajar con ceros y uno poniendo a cada uno la restricción que se desee en cada caso. En cuanto a este aspecto ambos SGBD no son compatibles.

Esta característica tendría un ***costo bajo*** de valor ***0.4*** pues de la misma forma en que se implementa una función de ceros y uno en Oracle podría realizarse en PostgreSQL.

**El tipo de dato CHAR:**

El estándar define character(n) como un tipo de caracter primario, donde *n* es un entero positivo. Este tipo de caracter permite almacenar cadenas de hasta *n* caracteres de longitud. Char (n) es un alias de character(n).

Un intento de almacenar una cadena más larga de lo especificado en el parámetro *n* resultará un error, a menos que el exceso de los caracteres esté dado por los espacios que contiene la cadena. En este caso la cadena será reducida a la longitud máxima.

Si la cadena que se almacena es más corta que la longitud declarada, será completada con espacios hasta llegar a dicha longitud.

PostgreSQL no cumple con la norma en parte, elimina los espacios en blancos contenidos en la cadena CHAR antes de realizar la mayoría de funciones, operadores y comparaciones. Ejemplo de ello a la hora de ejecutar la función CHARACTER LENGHT y de concatenar dos o más cadenas.

Oracle sigue el estándar, con una pequeña excepción: no quita los espacios que excedan la longitud especificada CHAR, pero eleva una excepción.

En cuanto a este tipo de datos ambos gestores no son compatibles pues presentan determinadas excepciones con respecto al estándar.

Lo antes explicado le arrojaría a esta característica un ***costo medio*** de valor ***0.5*** pues para garantizar la correcta conversión de datos entre estos dos gestores de bases de datos, debe asegurarse de que los datos CHAR se compongan de cadenas bien formadas, o sea que no se introduzcan cadenas que contengan espacios en blanco, además garantizando que no se escriban cadenas con espacios en blanco no se afectaría la portabilidad.

**El tipo de dato TIMESTAMP:**

Este tipo de datos almacena el año, mes, día, hora, minuto, segundo, hasta 6 décimas. Existe una extensión del estándar SQL que plantea el tipo de datos TIMESTAMP WITH TIME ZONE, que además almacena la zona horaria.

Ejemplos literales de TIMESTAMP:

- TIMESTAMP '2003-07-29 13:19:30'
- TIMESTAMP '2003-07-29 13:19:30.5'

Ejemplos de TIMESTAMP WITH TIME ZONE literales:

- TIMESTAMP '2003-07-29 13:19:30+02:00'
- TIMESTAMP '2003-07-29 13:19:30.5+02:00'

PostgreSQL cumple con el estándar, con la excepción de que en algunos casos TIMESTAMP '2003-08-23 01:02:03 +02:00' es interpretada como TIMESTAMP WITH**OUT** TIME ZONE (descartando la parte +02:00') no como un valor TIMESTAMP WITH TIME ZONE como se plantea en el estándar. Oracle cumple completamente con el estándar. En cuanto a esta característica ambos SGBD son generalmente compatibles pues ambos cumplen con lo estandarizado, aunque en el caso de PostgreSQL pudieran existir algunas incongruencias.

Esta característica tendría un ***costo bajo*** de valor ***0.4*** debido a que los cambios son mínimos y no en todas las ocasiones pues solo en algunas ocasiones en PostgreSQL el TIMESTAMP WITH TIME ZONE es tratado como un TIMESTAMP WITH**OUT** TIME ZONE eliminado la zona horaria, pues generalmente se hace de igual forma en ambos gestores.

**Character_LENGHT:**

La norma plantea que el argumento puede ser de tipo CHAR o VARCHAR, retornando NULL si la entrada es NULL. PostgreSQL cumple con el estándar, aunque hay que tener en cuenta que el comportamiento de CHARACTER_LENGTH con respecto a los valores CHAR ha cambiado entre versiones 7.4 y 8.0 de este gestor. Oracle, por su parte, no tiene la función CHARACTER_LENGTH proporcionando en su lugar la función LENGTH. Ambos SGBD no son compatibles en cuanto a esta función pues Oracle ofrece otra con la misma funcionalidad.

En el caso de esta característica se le otorga un ***costo alto*** de valor ***1*** pues no existe forma común y sencilla de realizar la misma operación mediante una misma función para ambos.

**SUBSTRING:**

La norma define 2 variantes de la función SUBSTRING:

1. Para cumplir con el sistema central de SQL, el DBMS debe ser compatible con una función SUBSTRING ordinaria, que extrae los caracteres de una cadena: SUBSTRING ( input FROM start-position [FOR length ] )
2. El resultado es NULL si alguno de los argumentos es NULL.

El DBMS, opcionalmente, puede ofrecer una variante de expresión regular de SUBSTRING:

- SUBSTRING ( input SIMILAR pattern ESCAPE escape-char )

PostgreSQL ofrece 3 variantes de la función SUBSTRING:

1. SUBSTRING Ordinario: Igual a la variante SUBSTRING ordinaria de la norma.
2. POSIX SUBSTRING: Expresión regular cuya sintaxis es: SUBSTRING (input FROM pattern-string).
3. Clasificación de estilo SQL SUBSTRING, expresión regular: la sintaxis es: SUBSTRING (input FROM pattern-string FOR escape-char).

Oracle no ofrece función SUBSTRING estándar, proporciona en su lugar:

- SUBSTR (input, start-pos [, length])

En cuanto a esta característica ambos SGBD no son compatibles.

Esta característica tendría un ***costo alto*** de valor ***1*** pues no existe similitud alguna entre las sentencias utilizadas para realizar esta función de forma tal que se realice de igual forma para ambos gestores.

**REPLACE:**

Por REPLACE se entiende, la función de una cadena que busca una cadena de origen (pajar) de las apariciones de una cadena que se sustituye (aguja) y la sustituye por una nueva cadena (de sustitución). El estándar no menciona la función REPLACE, parece haber surgido un estándar de facto sobre esta función:

- REPLACE (haystack: string, needle: string, replacement: string)

Esta función puede ser útil para corregir errores de ortografía y de otras situaciones ejemplo:

- UPDATE tablename SET fullname=REPLACE (fullname, ‘Jeo ','Joe ').

PostgreSQL al igual que Oracle sigue el estándar de facto. En este caso ambos SGBD son totalmente compatibles.

En el caso de esta característica debido a que no existe diferencia alguna en la forma de realizar esta operación en ambos SGBD el ***costo*** sería ***0***.

**Hora local**

Es muy importante en una base de datos tener el valor de fecha y hora actualizadas, pues en ellas se proporcionan un número de funciones que devuelven valores relacionados con la fecha y hora actuales. Además nos permite conocer la hora de inicio de una instrucción, el instante en que una función es llamada y resulta muy útil a la hora de programar las salvas automáticas de los datos. Según el estándar la marca de tiempo actual se recupera con la función LOCALTIMESTAMP. Esta característica no afecta la

portabilidad pues PostgreSQL al igual que Oracle siguen el estándar por lo que posee un ***costo*** de valor ***0***.

**Concatenación.**

La concatenación es, en general, el acto de unir, enlazar o combinar dos o más cadenas de caracteres en una única cadena. Según el estándar la concatenación de dos cadenas se realiza con el operador | |.

Ejemplo:

texto1 | | texto2

Si al menos un operando es NULL, el resultado es NULL. PostgreSQL sigue el estándar. Mientras que Oracle sigue el estándar, en parte. Como Oracle interpreta NULL como la cadena vacía, no devuelve NULL si un operando es NULL. En cuanto a este aspecto PostgreSQL y Oracle son parcialmente compatibles a la hora de realizar la concatenación, aunque ambos emplean el mismo operador que establece el estándar, se diferencian en cuanto a la presencia de un operador NULL en la concatenación como se explicó anteriormente. Por este motivo se ve afectada la portabilidad.

Esta característica tendría un ***costo medio*** de valor ***0.5*** porque con una ligera validación donde no se admita el uso de valores NULL en la concatenación, se lograría una forma común de realizar dicha operación.

**La restricción UNIQUE**

Las restricciones UNIQUE se emplean para garantizar que no se escriban valores duplicados en columnas de una tabla que no forman parte de una clave principal. En una tabla se pueden definir varias restricciones UNIQUE. Las restricciones UNIQUE admiten valores NULL, sin embargo, sólo se admite un valor NULL por columna. Cuando se crea una tabla se puede crear una restricción UNIQUE como parte de la definición de esa tabla. Si ya existe una tabla, se puede agregar una restricción UNIQUE, siempre que la columna o combinación de columnas que forman la restricción UNIQUE contenga sólo valores únicos. Una tabla puede contener varias restricciones

UNIQUE. (microsoft, 2008). El estándar plantea que como indica el nombre de la restricción, un conjunto de columna (s) con una restricción UNIQUE sólo puede contener una única combinación de los valores que estas posean. Una columna o un conjunto de columnas, que está sujeta a una restricción UNIQUE también deben estar sujetos a una restricción no NULL, a menos que el sistema de gestión de bases de datos implementa un facultativo "NULL permitido". PostgreSQL al igual que Oracle sigue el estándar por lo que se puede decir que ambos SGBD son totalmente compatibles en cuanto a este aspecto, favoreciendo así la portabilidad en caso de una migración por lo que el ***costo*** de esta es de valor ***0***.

**Generación automática de claves.**

A veces es útil que el sistema de gestión de bases de datos maneje la generación de claves, aunque esto pudiera traer algunos inconvenientes. Según el estándar el atributo GENERATED... AS IDENTITY se emplea para la generación de claves, que puede ser declarado para ciertos tipos de datos ejemplos:

- CREATE TABLE tablename ( tablename_id INTEGER GENERATED **ALWAYS** AS IDENTITY...)
- CREATE TABLE tablename ( tablename_id INTEGER GENERATED **BY DEFAULT** AS IDENTITY ...)

El atributo IDENTITY otorga valores en orden creciente, posiblemente con "agujeros" (..., 3, 4, 7,...). La columna de una tabla declarada con este atributo será declarada llave primaria. En los ejemplos anteriores se puede ver que estos difieren en el comando anterior a la cláusula IDENTITY:

- Cuando se especifica ALWAYS, el usuario no puede especificar un valor identificativo para la columna por lo que el sistema de gestión de bases de datos garantiza la asignación exitosa de un valor único para la columna.
- Cuando se especifica BY DEFAULT, el usuario puede especificar manualmente el valor identificativo para una fila.

PostgreSQL no es compatible con atributo GENERATED... AS IDENTITY del estándar, para este caso PostgreSQL emplea el atributo SERIAL ejemplo: TableName CREATE TABLE ( SERIAL tablename_id,...)

SERIAL se usa para la creación y uso de una secuencia que permita crear enteros únicos para una columna.

Oracle por su parte no admite el atributo GENERATED... AS IDENTITY del estándar. Para lograr esto en Oracle se crea una secuencia según se desee y luego es usada en un disparador asociado a la tabla en cuestión. Esto creará un valor único para la columna que se desee que sea llave primaria.

Esta característica afecta totalmente la portabilidad pues ambos SGBD no son compatibles en cuanto a la generación de claves automáticas, pues cada uno de ellos lo realiza de forma diferente e incluso ninguno es compatible con el estándar.

Esta característica tiene un ***costo alto*** de valor ***1*** pues es muy difícil por no decir casi imposible lograr que se generen estos identificadores de igual forma en ambos gestores pues en cada uno de ellos se hace de manera diferente y adaptado a las características del gestor.

**TRUNCATE TABLE**

En muchas ocasiones a la hora de eliminar todas las filas de una tabla grande la cláusula DELETE no ofrece gran rapidez al ejecutar esta acción. Para resolver este problema varios sistemas de gestión de bases de datos aplican una operación llamada TRUNCATE que ofrece mayor rapidez al implicar poca o ninguna actividad de registro de transacciones. La desventaja es que TRUNCATE solo sirve cuando se quiere eliminar absolutamente todos los registros, ya que no se permite la cláusula WHERE.

El estándar SQL define la declaración TRUNCATE TABLE *tablename* (nueva en SQL: 2008) como:

Eliminar todas las filas de una tabla, sin causar ninguna acción activa.

El estándar no especifica si:

- TRUNCATE TABLE debe permitirse en una transacción que involucra otras declaraciones o no.
- TRUNCATE TABLE debe implicar un COMMIT inmediato, o no.

PostgreSQL: Sigue el estándar, pues en PostgreSQL, TRUNCATE TABLE permite que una transacción pueda involucrar otras operaciones y no implica una operación COMMIT inmediata, mientras que por su parte Oracle sigue el estándar.

Ambos sistemas gestores de bases de datos son compatibles para esta característica, por lo que esta favorece la portabilidad, aunque hay que tener en cuenta que PostgreSQL admite el uso de TRUNCATE TABLE en una transacción que puede involucrar otras operaciones, mientras que en Oracle no se tiene seguridad de que se pueda hacer esto pues este sigue al estándar el cual no plantea si esto es posible.

En el caso de esta característica debido a que no existe diferencia alguna en la forma de realizar esta operación en ambos SGBD el ***costo*** sería ***0***.

**Procedimientos con comandos.**

Las siguientes características no son necesariamente operaciones de SQL, sino más bien cómo se desarrolla una serie de operaciones en la interfaz de líneas de comandos. Solo se analizarán algunas de estas características pues en la UCI no se emplean estas interfaces.

**Recopilar estadísticas manualmente de una tabla.**

Esta característica no está estandarizada. En PostgreSQL se realiza mediante la sentencia ANALYZE tablename, ANALYZE recopila y almacena estadísticas sobre el contenido de una tabla de la base de datos, las cuales se emplean para la ejecución más eficiente de las consultas. Oracle ofrece para recopilar las estadísticas de una tabla la utilización del comando ANALYZE igualmente. De esta forma esta característica favorece la portabilidad ya que ambos gestores emplean el mismo comando.

En el caso de esta característica debido a que no existe diferencia alguna en la forma de realizar esta operación en ambos SGBD el ***costo*** sería ***0***.

**Obtener una descripción de consulta**

Esta característica no está estandarizada. PostgreSQL lo realiza mediante la sentencia EXPLAIN <query>, EXPLAIN es un comando que muestra el plan de ejecución para una instrucción, que escanea la tabla (s) que forman parte de esta, obteniéndose el tiempo en devolver las filas en cuestión. Esto es muy útil a la hora de estimar el coste total de la instrucción. En Oracle se realiza mediante la sentencia EXPLAIN PLAN FOR <query>, la instrucción EXPLAIN PLAN se utiliza para determinar el plan de ejecución de base de datos Oracle al ejecutar una instrucción SQL especificada. Este comando inserta una fila que describe cada paso del plan de ejecución en una tabla especificada. Esta declaración también determina el costo de la ejecución de la instrucción. Ambos no son compatibles para esta característica pues usan instrucciones diferentes. Destacar que esta característica no está estandarizada lo que explica que ambos la manejen de forma diferente, haciendo nula la portabilidad en caso de una migración por lo que el ***costo*** sería de un valor ***alto*** igual a ***1***.

**Tiempo de una consulta.**

Esta característica no se encuentra en el estándar. PostgreSQL lo realiza mediante el comando *timing*. Este comando muestra cuánto tarda cada comando SQL en milisegundos. Oracle por su parte lo realiza mediante el comando SET TIMING ON. Este comando controla la visualización de las estadísticas de tiempo. Esta característica afecta la portabilidad pues ambos gestores la tratan de forma diferente debido a que esta característica no está estandarizada por lo que el ***costo*** sería de un valor ***alto*** igual a ***1***.

## 2.2. Portabilidad entre versiones de PostgreSQL.

La portabilidad entre las versiones de PostgreSQL, según las investigaciones realizadas, no es preocupación cuando se migra desde una versión a una posterior. En casos de que se desarrolle una aplicación con la última versión y a la hora de desplegar, el cliente no posea las condiciones necesarias para adoptar este sistema, entonces si el tema de la portabilidad sería una preocupación. Muchas características que se implementan a partir de determinada versión, no se encuentran en versiones anteriores, lo cual sería un problema para migrar de versiones superiores a las anteriores.

El costo de cada una de estas características va a ser un ***costo alto*** de valor ***1*** pues en todos los casos son características nuevas en determinada versión y que no existen en otras versiones anteriores por tanto la complejidad de realizar esa determinada operación en esas versiones anteriores es alta.

**Expresión de tabla común con recursividad. Common tables expressions (CTE) (WHITH RECUSRIVE)**

Una expresión de tabla común (CTE, por sus siglas en inglés) ofrece la importante ventaja de poder hacer referencia a sí mismo, creando así una CTE recursiva. Una CTE recursiva es aquella en la que una CTE inicial se ejecuta varias veces para devolver subconjuntos de datos hasta que se obtiene el conjunto de resultados completo.

Una CTE recursiva puede simplificar el código necesario para ejecutar una consulta recursiva dentro de un SELECT, INSERT, UPDATE, DELETE o CREATE VIEW.

Esta característica no se encuentra en versiones anteriores a la 8.3, incluyendo la misma, por lo que se puede decir entre estas versiones y la 8.4 no hay compatibilidad en cuanto a este aspecto.

**Comando DROP**

Este comando elimina un objeto de la base de datos. Puede ser una tabla, vista, índice, trigger, función, procedimiento o cualquier otro objeto que el motor de la base de datos soporte. Esta característica se implementa a partir de la versión 8.2 de PostgreSQL, por lo que esta versión y las superiores a esta no son compatibles en cuanto a este aspecto con las versiones anteriores a la misma.

**Uso del RETURNING combinado con INSERT/UPDATE/DELETE**

La cláusula INSERT inserta nuevos registros en una tabla. Se puede insertar una o más filas. UPDATE permite la actualización de cualquier columna o las columnas de la tabla especificada y DELETE permite eliminar elementos de la tabla especificada. Por su parte la cláusula RETURNING nos sirve para capturar datos de forma fácil, luego de realizar un Insert, Update y/o Delete.
Esta característica comienza a ser usada a partir de la versión 8.2 de PostgreSQL. Por lo que entre versiones anteriores a la 8.2 y las superiores a la misma incluyéndola no hay compatibilidad en cuanto a este aspecto afectando la portabilidad entre ellas en caso de una posible migración.

**Utilización del autovacuum**

VACUUM (vacío) es un comando SQL que limpia y opcionalmente analiza una base de datos. Trabaja con las tuplas muertas, o sea las tuplas que son eliminadas u obsoletas por una actualización no son físicamente removidas de su tabla; permanecen presentes hasta que se hace un VACUUM quedando totalmente eliminadas. Por lo tanto, es necesario hacer un VACUUM periódicamente, especialmente sobre tablas actualizadas frecuentemente.
PostgreSQL tiene un Autovacuum opcional, cuyo objetivo es automatizar la ejecución de VACUUM. Cuando se activa, los controles Autovacuum en las tablas que se ha insertado, actualizado o eliminado algún dato, estos

analizan automáticamente las tablas cuando se carga la base de datos por primera vez quedando todos los datos actualizados. PostgreSQL comienza a utilizar este comando a partir de la versión 8.1, en versiones anteriores a este no se encuentra disponible afectando así la portabilidad en caso de una migración a estas versiones anteriores.

**Cláusula ORDER BY**

La cláusula ORDER BY hace que las filas que resultan de una determinada consulta se ordenen de acuerdo con una expresión especificada, o sea que pueden ordenarse por más de una expresión. Si dos filas son iguales de acuerdo a la primera expresión definida, se comparan de acuerdo a la expresión siguiente y así sucesivamente. Si son iguales de acuerdo a todas las expresiones especificadas, se devuelven en un orden que determine la aplicación.

Esta cláusula permite ordenar de forma ascendente y descendente. Opcionalmente se puede agregar la palabra clave ASC (ascendente) o DESC (descendente) después de cualquier expresión en la cláusula ORDER BY. Si no se especifica, se asume por defecto un ordenamiento ascendente.

También se tienen en cuenta los valores NULL mediante las palabras claves NULLS LAST para ordenar los valores nulos después de los no nulos y NULLS FIRST para ordenarlos antes. Si no se especifica ninguno, el comportamiento por defecto es NULLS LAST.

El uso de esta cláusula es permitido en la versión 8.3 de PostgreSQL y en las superiores a esta. En caso de que se necesite migrar a una inferior se vería afectada la portabilidad pues estas no son compatibles con esta cláusula.

**Uso de tipos de datos compuestos en matrices**

Un tipo de dato compuesto representa la estructura de una fila o registro, es esencialmente sólo una lista de nombres de campos y sus tipos de datos. PostgreSQL permite que los tipos de datos compuestos se utilicen en

muchos casos de la misma manera que utiliza los tipos de datos simples. Por ejemplo, una columna de una tabla puede ser declarada de un tipo de datos compuesto.

PostgreSQL permite que en cualquier matriz declarada por un usuario se pueda declarar un tipo de dato compuesto. El uso de este tipo de dato afectaría la portabilidad pues esta característica solo es usada en la versión 8.3 de PostgreSQL y sus sucesoras, no siendo posible una migración a una versión inferior.

**Tipos de datos ENUM**

Enumerado (enum) son tipos de datos que declaran una enumeración para determinada característica que son estáticas para el entorno donde se definen. Son una forma de crear tipos de datos con referencias internas. Un ejemplo de un tipo de enumeración podrían ser los días de la semana.

*CREATE TYPE días ENUM AS*
*('lunes','martes','miercoles','jueves','viernes','sabado','domingo');*
*CREATE TABLE ejemplos (unsolodia día, variosdias día []);*
*INSERT INTO ejemplos VALUES ('lunes'::día, '{lunes, martes}'::día []);*

El empleo de estos tipos de datos es válido para la versión 8.3 de PostgreSQL y las superiores a esta, por lo que un sistema que utilice esta característica no sería portable a una versión inferior de las antes mencionadas.

**Tipos de datos UUID**

Los Identificadores Universalmente Únicos (UUID por sus siglas en inglés) definidos por la ISO / IEC 9834-8:2005, son generados por un algoritmo seguro para que sea muy poco probable que el mismo identificador sea generado por cualquier otro algoritmo. Estos identificadores proporcionan una gran garantía en la singularidad de generadores de secuencias, que deben ser únicos en la base de datos. La utilización de estos identificadores

atenta contra la portabilidad pues solo son permitidos a partir de la versión 8.3 de PostgreSQL.

**Sintaxis de fecha de la NORMA ISO 8601**

La norma ISO 8601 especifica la notación estándar utilizada para almacenar las fechas.

El estándar internacional para la notación de la fecha es: AAAA-MM-DD.

Donde AAAA es el año en el calendario gregoriano, MM es el mes en el año entre 01 (Enero) y 12 (Diciembre), y DD es el día del mes entre 01 y 31.

Las ventajas del estándar ISO 8601 con respecto a otras notaciones son:

- Es fácil procesar estas fechas sin tablas de meses auxiliares.
- Es fácil compararlas y ordenarlas.
- Independiente del idioma.
- No se confunde con otras notaciones.

Si se está empleando la versión 8.4 de PostgreSQL en el desarrollo de un sistema y se utiliza la norma explicada anteriormente no sería portable a una versión inferior pues ninguna de estas es compatible con la norma en cuestión.

**Uso de tipos de Datos Null en matrices.**

El empleo de tipos de datos NULL en las matrices afecta a la portabilidad pues esto solo es permitido en la versión 8.2 de PostgreSQL o una superior a esta.

**Tipo de datos XML.**

El tipo de datos XML se puede utilizar para almacenar datos XML. Un documento XML puede ser almacenado como una única columna en la base de datos y particionado en columnas y tablas en la base de datos ofreciendo una facilidad a los desarrolladores a la hora de traducir datos XML a una estructura relacional. Un sistema que utilice esta característica no sería

totalmente portable, pues solo se permite el uso de la misma a partir de la versión 8.3 de PostgreSQL.

**Cláusula RETURN QUERY EXECUTE**

Esta cláusula permite devolver los resultados generados por una consulta dinámicamente. El empleo de la misma podría traer problemas de portabilidad pues solo es válido su uso en la versión 8.4 de PostgreSQL.

**Cláusula TRUNCATE**

TRUNCATE elimina rápidamente todas las filas de un conjunto de tablas, la cláusula DELETE también realiza esta función, pero lo hace de una forma más lenta que TRUNCATE. El uso de esta cláusula afecta la portabilidad pues solo es usada por la versión 8.4 de PostgreSQL.

**Expresión condicional CASE en Pl/pgsql**

Pl/pgsql permite el uso de CASE como una expresión condicional que permiten ejecutar comandos alternativos basados en determinadas condiciones. CASE proporciona ejecución condicional basada en la verdad de expresiones lógicas. Cada expresión se evalúa (una vez), si se encuentra una coincidencia, entonces las declaraciones que se encuentran dentro del CASE se ejecutan y luego el control pasa a la siguiente instrucción después de END CASE ( fin de la instrucción CASE). Si no hay ninguna expresión verdadera, se ejecutan las instrucciones que le siguen al END CASE. Esta característica afecta a la portabilidad en caso de una migración de la versión 8.4 de PostgreSQL a una inferior, pues ninguna de estas utiliza esta expresión condicional.

**Manejo de excepciones en PL/pgsql**

Cuando se produce un error a la hora de ejecutar una función en PL/ pgsql esta se anula además de las transacciones circundantes. Estos errores se pueden capturar y darle tratamiento con una cláusula de excepción

permitiendo que no se interrumpan las funciones que se estén desarrollando, además de controlar y prevenir los errores que pueden surgir. Esta característica afectaría la portabilidad en caso de una migración de la versión 8.0 de PostgreSQL o una superior, a la versión 7.4 pues esta última no emplea dicha característica.

**Creación de lenguajes**

Mediante la sentencia CREATE LANGUAGE, un usuario de PostgreSQL puede registrar un idioma o procedimiento nuevo en una base de datos PostgreSQL. Posteriormente, las funciones y los procedimientos que el usuario desee los puede definir en este nuevo lenguaje. El uso de esta sentencia atenta contra la portabilidad pues su uso solo es válido a partir de la versión 8.3 de PostgreSQL.

**Número de argumentos de una función**

Para las versiones 7.4 y 8.0 de PostgreSQL el límite de números de argumentos en una función era de 32, a partir de la versión 8.1 el número aumentó a 100, lo que provoca incompatibilidades entre ellas, afectando así a la portabilidad.

**Uso del RETURN QUERY en PL / pgsql**

Cuando se declara una función en PL / pgsql esta puede ser declarada para devolver varias columnas de una tabla. El comando RETURN QUERY añade los resultados de las consultas que recogen los datos de dichas columnas al resultado final de la función. Esta característica afecta a la portabilidad, pues este comando solo puede ser usado si se está utilizando la versión 8.3 de PostgreSQL o una superior a esta.

**Variables SQLSTATE y SQLERRM para PL / pgsql**

Dentro de un controlador de excepciones, la variable SQLSTATE contiene una lista de posibles códigos de error correspondientes a diferentes

excepciones que puedan ocurrir. La variable SQLERRM contiene el mensaje de error asociado con la excepción, permitiéndose un acceso sencillo al estado de error de SQL y el texto de error asociado, si ocurrió una excepción. El uso de estas variables surge en la versión 8.1 de PostgreSQL, lo puede afectar la portabilidad en caso de una migración a una versión anterior a la mencionada.

## 2.3. Métricas para evaluar portabilidad entre PostgreSQL y Oracle.

Según la investigación realizada, descrita en el epígrafe anterior, sobre algunas de las características fundamentales evaluadas para los SGBD PostgreSQL y Oracle. En este epígrafe se representan un grupo de métricas, que se definieron a partir de lo desarrollado en el epígrafe anterior, las cuales constituyen la base para medir la portabilidad entre ambos SGBD.

La medida en la que cada una de estas métricas afecte la portabilidad va a estar dada fundamentalmente por la medida en la que se use cada una de las características identificadas y el costo o criticidad de realizar un cambio de estas en el modelo de datos del proyecto en cuestión.

*Estructura de la representación de las métricas.*

La métrica representa la característica que puede afectar la portabilidad en el modelo. La precondición es una condición que debe cumplirse antes de realizar un análisis de la característica en cuestión. El costo no es más que el valor asignado a la característica de acuerdo con la complejidad de hacer la misma portable para ambos SGBD explicado en el epígrafe anterior.

| ***Métrica # 1*** | Uso de actualizaciones de vistas. |
|---|---|
| ***Precondición*** | No tiene |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 2*** | Uso de la cláusula LIKE. |
| ***Precondición*** | No tiene |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 3*** | Utilización del límite simple |
| ***Precondición*** | No tiene |
| ***Costo*** | **0.6** |

| | |
|---|---|
| ***Métrica # 4*** | Inserción múltiple de tuplas |
| ***Precondición*** | No tiene |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 5*** | Uso del tipo de datos BOOLEANO |
| ***Precondición*** | No tiene |
| ***Costo*** | **0.4** |

| | |
|---|---|
| ***Métrica # 6*** | Uso del tipo de datos CHAR |
| ***Precondición*** | No tiene |
| ***Costo*** | **0.5** |

| | |
|---|---|
| ***Métrica # 7*** | Uso del tipo de datos TIMESTAMP WITH TIME ZONE |
| ***Precondición*** | No tiene |
| ***Costo*** | **0.4** |

| | |
|---|---|
| ***Métrica # 8*** | Uso de la función CHARACTER_LENGTH |
| ***Precondición*** | No tiene |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 9*** | Uso de la función SUBSTRING |
| ***Precondición*** | No tiene |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 10*** | Concatenación de columnas que admitan valores nulos. |
| ***Precondición*** | No tiene |
| ***Costo*** | **0.5** |

| | |
|---|---|
| ***Métrica # 11*** | Generación automática de claves |
| ***Precondición*** | No tiene |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 12*** | Obtención de descripciones de consultas |
| ***Precondición*** | No tiene |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 13*** | Obtención del tiempo de consulta |
| ***Precondición*** | No tiene |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 14*** | Utilización de valores NULL en la expresión Order By. (NULL FIRST Y NULL LAST) |
| ***Precondición*** | Que se vaya a migrar para una versión inferior a la 8.3 de PostgreSQL. |
| ***Costo*** | **1** |

## 2.4. Métricas para evaluar portabilidad entre versiones de PostgreSQL.

| | |
|---|---|
| ***Métrica # 1*** | Uso de expresiones de tabla común con recursividad |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.4 de PostgreSQL y se vaya a migrar para cualquier versión anterior. |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 2*** | Utilización del comando Drop |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.2 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.2 |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 3*** | Uso del RETURNING combinado con INSERT/UPDATE/DELETE |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.2 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.2 |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 4*** | Uso del Autovacuum |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.1 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.1 |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 5*** | Empleo de la cláusula Order By |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.3 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.3 |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 6*** | Uso de tipos de datos compuestos en matrices |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.3 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.3 |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 7*** | Uso del tipo de datos Enum |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.3 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.3 |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 8*** | Uso de los Identificadores Universalmente Únicos (UUID) |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.3 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.3 |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 9*** | Uso de la Sintaxis de fecha de la NORMA ISO 8601 |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.4 y se vaya a migrar para cualquier versión anterior a la misma. |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 10*** | Uso de tipos de Datos Null en matrices. |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.2 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.2 |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 11*** | Uso del Tipo de datos XML. |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.3 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.3 |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 12*** | Uso de la cláusula RETURN QUERY EXECUTE |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.4 y se vaya a migrar para cualquier versión anterior a esta. |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 13*** | Uso de la cláusula TRUNCATE |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.4 y se vaya a migrar para cualquier versión anterior a esta. |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 14*** | Uso de la expresión condicional CASE en Pl/pgsql |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.4 y se vaya a migrar para cualquier versión anterior a esta. |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 15*** | Creación de lenguajes |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.3 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.3 |
| ***Costo*** | **1** |

| | |
|---|---|
| ***Métrica # 16*** | Número de argumentos de una función |
| ***Precondición*** | Que se encuentre trabajando en la versión 8.1 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.1 y además que el número de argumentos de al menos una función sea mayor que 32. |
| ***Costo*** | **1** |

| ***Métrica # 17*** | Uso del RETURN QUERY en PL / pgsql |
|---|---|
| ***Precondición*** | Que se encuentre trabajando en la versión 8.3 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.3 |
| ***Costo*** | **1** |

| ***Métrica # 18*** | Uso de variables SQLSTATE y SQLERRM para PL / pgsql |
|---|---|
| ***Precondición*** | Que se encuentre trabajando en la versión 8.1 o alguna superior a esta de PostgreSQL y se vaya a migrar para cualquier versión anterior a la 8.1 |
| ***Costo*** | **1** |

## 2.5. Conclusiones

En el presente capítulo se realizó un análisis exhaustivo de las características que afectan la portabilidad entre los SGBD PostgreSQL y Oracle así como entre las diferentes versiones de PostgreSQL. A partir de este análisis se establecieron las métricas que permitirán medir la portabilidad de los datos y se le definió a cada una de estas el costo que tendrían al realizarle modificaciones al modelo de datos para que este sea portable.

A partir de las métricas establecidas se decidió elaborar una herramienta que permita la evaluación de la portabilidad de los datos entre los SGBD PostgreSQL y Oracle y entre las versiones de la 7.4 a la 8.4 de PostgreSQL.

# 3. Descripción y validación de la herramienta

En este capítulo se le da cumplimiento al objetivo principal de la presente investigación con la elaboración de una herramienta que permita evaluar la portabilidad en la gestión de los datos de las soluciones informáticas desarrolladas en la UCI, basada en las métricas desarrolladas en el capítulo anterior. Se describe la elaboración de dicha herramienta donde se expondrá la estructura, la forma de uso y las fórmulas empleadas en la misma, además se hace una evaluación del funcionamiento y los resultados de la herramienta.

Las métricas establecidas para realizar la evaluación de la portabilidad a través de la herramienta desarrollada fueron validadas previamente por personas con conocimiento y experiencia en el tema, encontrándose en los anexos los documentos que avalan esta validación.

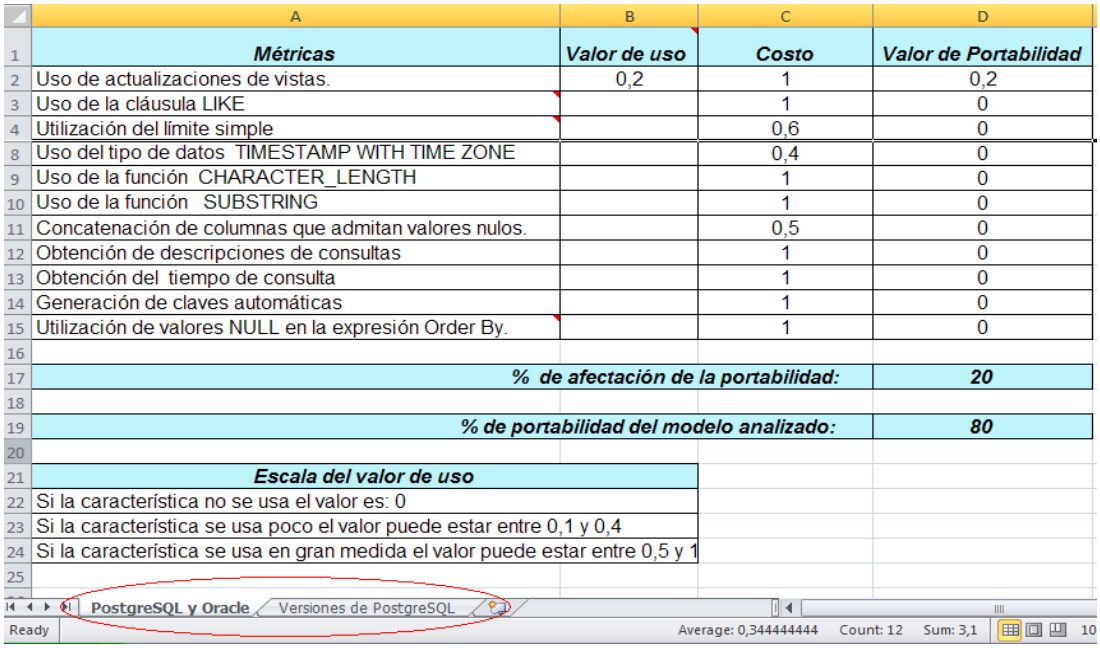
Según la opinión de los expertos que analizaron las mismas, estas pueden ser utilizadas para evaluar la portabilidad, obteniéndose de las mismas resultados concretos y fundamentales para conocer en qué medida es portable una base de datos.

## 3.1. Descripción de la Herramienta.

La herramienta que permitirá evaluar la portabilidad está hecha a modo de cuestionario debido a que en la UCI los modelos de datos en muchas ocasiones llevan código SQL incrustado en la capa de acceso a datos y llevaría mucho tiempo lograr una herramienta que analice todo el código de un modelo, fue por esto que se desarrolló una herramienta sobre Microsoft Excel ya que en este los cálculos son muy exactos y precisos lo que contribuye al objetivo de esta herramienta que es dar al cliente un porciento final de cuan portable es el modelo de datos que posee.

La herramienta que lleva por nombre ***Evaluador de Portabilidad*** (fig. 1) consta de dos hojas, la primera destinada a evaluar la portabilidad de los datos entre los SGBD PostgreSQL y Oracle y la segunda entre las versiones

de la 7.4 a la 8.4 de PostgreSQL. Ambas hojas presentan la misma estructura (fig. 2).



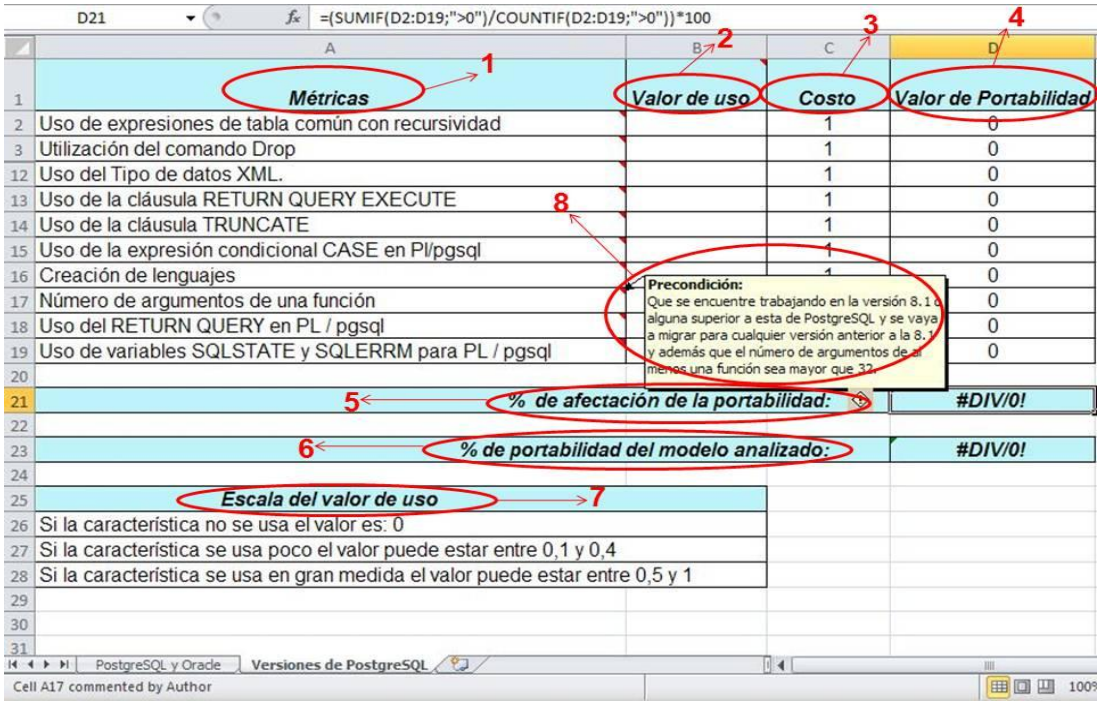
| | Métricas | Valor de uso | Costo | Valor de Portabilidad |
|---|---|---|---|---|
| 2 | Uso de actualizaciones de vistas. | 0,2 | 1 | 0,2 |
| 3 | Uso de la cláusula LIKE | | 1 | 0 |
| 4 | Utilización del límite simple | | 0,6 | 0 |
| 8 | Uso del tipo de datos TIMESTAMP WITH TIME ZONE | | 0,4 | 0 |
| 9 | Uso de la función CHARACTER_LENGTH | | 1 | 0 |
| 10 | Uso de la función SUBSTRING | | 1 | 0 |
| 11 | Concatenación de columnas que admitan valores nulos. | | 0,5 | 0 |
| 12 | Obtención de descripciones de consultas | | 1 | 0 |
| 13 | Obtención del tiempo de consulta | | 1 | 0 |
| 14 | Generación de claves automáticas | | 1 | 0 |
| 15 | Utilización de valores NULL en la expresión Order By. | | 1 | 0 |
| 17 | % de afectación de la portabilidad: | | | 20 |
| 19 | % de portabilidad del modelo analizado: | | | 80 |

| | Escala del valor de uso |
|---|---|
| 22 | Si la característica no se usa el valor es: 0 |
| 23 | Si la característica se usa poco el valor puede estar entre 0,1 y 0,4 |
| 24 | Si la característica se usa en gran medida el valor puede estar entre 0,5 y 1 |

PostgreSQL y Oracle | Versiones de PostgreSQL

Ready — Average: 0,344444444 Count: 12 Sum: 3,1

Fig1: Evaluador de Portabilidad




| | Métricas | Valor de uso | Costo | Valor de Portabilidad |
|---|---|---|---|---|
| 2 | Uso de expresiones de tabla común con recursividad | | 1 | 0 |
| 3 | Utilización del comando Drop | | 1 | 0 |
| 12 | Uso del Tipo de datos XML. | | 1 | 0 |
| 13 | Uso de la cláusula RETURN QUERY EXECUTE | | 1 | 0 |
| 14 | Uso de la cláusula TRUNCATE | | 1 | 0 |
| 15 | Uso de la expresión condicional CASE en Pl/pgsql | | 1 | 0 |
| 16 | Creación de lenguajes | | 1 | 0 |
| 17 | Número de argumentos de una función | | 1 | 0 |
| 18 | Uso del RETURN QUERY en PL / pgsql | | 1 | 0 |
| 19 | Uso de variables SQLSTATE y SQLERRM para PL / pgsql | | 1 | 0 |
| 21 | % de afectación de la portabilidad: | | | #DIV/0! |
| 23 | % de portabilidad del modelo analizado: | | | #DIV/0! |

| | Escala del valor de uso |
|---|---|
| 26 | Si la característica no se usa el valor es: 0 |
| 27 | Si la característica se usa poco el valor puede estar entre 0,1 y 0,4 |
| 28 | Si la característica se usa en gran medida el valor puede estar entre 0,5 y 1 |

Fig. 2: Principales funcionalidades de la herramienta.

1. *Métricas:* En esta columna están expuestas las métricas obtenidas en el capítulo 2.
2. *Valor de uso:* En esta columna el usuario le otorgará un valor entre 0 y 1 a cada métrica, este valor no es más que la medida en la que se usa la característica implícita en cada una de ellas en el sistema que se esté evaluando. En la celda que lleva el nombre de la columna en cuestión está insertado un comentario que referencia al usuario a la *Escala del Valor de Uso,* la cual especifica qué valores se le pueden otorgar de acuerdo al uso que se le dé a la característica:
   No se usa: 0
   Se usa poco: de 0.1 a 0.4
   Se usa en gran medida: de 0.5 a 1.
3. *Costo:* El costo no es más que la criticidad de realizar un cambio en las características que representan cada métrica en el modelo de datos del proyecto en cuestión para que sea portable. Este costo es un valor entre 0,1 y 1 que fue definido en el capítulo anterior, el mismo no podrá ser modificado por el usuario ya que es fijo para cada una de las métricas expuestas. En el caso de las métricas para la portabilidad entre versiones de PostgreSQL el costo va a tomar dos únicos valores, 0 y 1, será 0 cuando no se cumpla con la precondición establecida para la métrica y 1 cuando si se cumpla con esta, pues la característica afectaría la portabilidad del modelo solo cuando se cumpla con la precondición establecida para esta, en caso contrario no atenta contra la portabilidad del modelo.
4. *Valor de portabilidad:* El valor de portabilidad consiste en que cada métrica tendrá asignado un valor que representa la medida en la que cada una de estas afecta la portabilidad, este valor no es más que la multiplicación del valor de uso por el costo para cada métrica en cuestión. Este valor será generado por la herramienta mediante una fórmula que realiza la multiplicación automáticamente.

5. *% de afectación de la portabilidad:* En esta fila se obtiene un valor que representará el porcentaje en que afectan todas las métricas en conjunto al sistema en cuestión, el mismo se obtiene sumando todas las celdas que contengan valores mayores que 0 de la columna Valor de Portabilidad, se dividen entre la cantidad de celdas que igualmente contengan valores mayores que cero y el resultado se multiplica por 100. Este valor será generado automáticamente por la herramienta con la ayuda de las ventajas que proporciona la herramienta Excel de Office.
6. *% de portabilidad del modelo analizado:* En esta fila se muestra el resultado general de portabilidad del modelo al que se le esté evaluando la portabilidad, para lo cual se le resta al 100% el % de afectación de la portabilidad.
7. *Escala del valor de uso: C*ontiene los valores que se le puede asignar a la característica de acuerdo con el uso que se le dé en el modelo de datos.
8. *Precondición y breve explicación de la característica:* Las características implícitas en dichas métricas que pudieran ocasionar algún tipo de duda al usuario a la hora de ser analizadas presentan un comentario en la celda que las contiene con una breve explicación de las mismas, además en dicho comentario aparece también la precondición que debe ser analizada antes de evaluar la característica en caso de tener precondición la misma.

Es importante resaltar que todos los valores que no pueden ser modificados en la herramienta están protegidos contra escritura por lo que el cliente solo podrá modificar los valores que deben ser introducidos por este.

## 3.2. Evaluación de la herramienta.

Para la evaluación de la herramienta desarrollada, se hizo una prueba piloto de la misma en algunos de los proyectos que se desarrollan en la UCI, los cuales se encuentran entre los que fueron encuestados al inicio de la

investigación, Sistema de Gestión Fiscal (SGF), Sistema Autónomo de Registros y Notarías (SAREN) y Sistema de Gestión Penitenciaria (Prisiones).

En el proyecto SGF se trabaja con la versión 8.2 de PostgreSQL y se están tomando precauciones pensando en la posibilidad de que el cliente no posea las condiciones necesarias para soportar esta versión de PostgreSQL y tengan que migrar toda la información para una versión anterior a la misma.

Además se evaluó la información para determinar la posibilidad de migrar hacia el SGBD Oracle.

Los resultados obtenidos en la herramienta indican que la información correspondiente a la base de datos del proyecto SGF posee un 10% de afectación y un 90% de portabilidad hacia versiones anteriores a la 8.2 de PostgreSQL, siendo esta no portable a estas versiones. (Fig. 3.1)

| | A | B | C | D |
|---|---|---|---|---|
| 1 | ***Métricas*** | ***Valor de uso*** | ***Costo*** | ***Valor de Portabilidad*** |
| 2 | Uso de expresiones de tabla común con recursividad | 0 | 0 | 0 |
| 3 | Utilización del comando Drop | 0 | 1 | 0 |
| 4 | Uso del RETURNING combinado con INSERT/UPDATE/DELETE | 0 | 1 | 0 |
| 5 | Uso del Autovacuum | 0 | 1 | 0 |
| 6 | Empleo de la cláusula Order By | 0 | 0 | 0 |
| 7 | Uso de tipos de datos compuestos en matrices | 0 | 0 | 0 |
| 8 | Uso del tipo de datos Enum | 0 | 0 | 0 |
| 9 | Uso de los Identificadores Universalmente Únicos (UUID) | 0 | 0 | 0 |
| 10 | Uso de la Sintaxis de fecha de la NORMA ISO 8601 | 0 | 0 | 0 |
| 11 | Uso de tipos de Datos Null en matrices | 0 | 1 | 0 |
| 12 | Uso del Tipo de datos XML. | 0 | 1 | 0 |
| 13 | Uso de la cláusula RETURN QUERY EXECUTE | 0,1 | 0 | 0 |
| 14 | Uso de la cláusula TRUNCATE | 0 | 0 | 0 |
| 15 | Uso de la expresión condicional CASE en Pl/pgsql | 0 | 0 | 0 |
| 16 | Creación de lenguajes | 0 | 0 | 0 |
| 17 | Número de argumentos de una función | 0,1 | 1 | 0,1 |
| 18 | Uso del RETURN QUERY en PL / pgsql | 0,1 | 0 | 0 |
| 19 | Uso de variables SQLSTATE y SQLERRM para PL / pgsql | 0 | 1 | 0 |
| 20 | | | | |
| 21 | ***% de afectación de la portabilidad:*** | | | ***10*** |
| 22 | | | | |
| 23 | ***% de portabilidad del modelo analizado:*** | | | ***90*** |

PostgreSQL y Oracle | Versiones de PostgreSQL

Ready

Fig. 3.1 Resultados de la validación en el proyecto SGF.

En cuanto a la portabilidad hacia otro gestor, en este caso hacia Oracle, la información contenida en la base de datos del proyecto SGF resultó tener un 18,80% de afectación de la portabilidad, resultando ser un 81.20% portable al SGBD Oracle. (Fig. 3.2)

| | A | B | C | D |
|---|---|---|---|---|
| 1 | ***Métricas*** | ***Valor de uso*** | ***Costo*** | ***Valor de Portabilidad*** |
| 2 | Uso de actualizaciones de vistas. | 0 | 1 | 0 |
| 3 | Uso de la cláusula LIKE | 0 | 1 | 0 |
| 4 | Utilización del límite simple | 0 | 0,6 | 0 |
| 5 | Inserción múltiple de tuplas | 0 | 1 | 0 |
| 6 | Uso del tipo de datos BOOLEANO | 0,3 | 0,4 | 0,12 |
| 7 | Uso del tipo de datos CHAR | 0,4 | 0,5 | 0,2 |
| 8 | Uso del tipo de datos TIMESTAMP WITH TIME ZONE | 0,3 | 0,4 | 0,12 |
| 9 | Uso de la función CHARACTER_LENGTH | 0 | 1 | 0 |
| 10 | Uso de la función SUBSTRING | 0 | 1 | 0 |
| 11 | Concatenación de columnas que admitan valores nulos. | 0 | 0,5 | 0 |
| 12 | Obtención de descripciones de consultas | 0 | 1 | 0 |
| 13 | Obtención del tiempo de consulta | 0 | 1 | 0 |
| 14 | Generación de claves automáticas | 0,4 | 1 | 0,4 |
| 15 | Utilización de valores NULL en la expresión Order By. | 0,1 | 1 | 0,1 |
| 16 | | | | |
| 17 | ***% de afectación de la portabilidad:*** | | | ***18,80*** |
| 18 | | | | |
| 19 | ***% de portabilidad del modelo analizado:*** | | | ***81,20*** |
| 20 | | | | |
| 21 | ***Escala del valor de uso*** | | | |
| 22 | Si la característica no se usa el valor es: 0 | | | |
| 23 | Si la característica se usa poco el valor puede estar entre 0,1 y 0,4 | | | |
| 24 | Si la característica se usa en gran medida el valor puede estar entre 0,5 y 1 | | | |

PostgreSQL y Oracle | Versiones de PostgreSQL

Fig. 3.2 Resultados de la validación en el proyecto SGF.

Los mayores problemas de portabilidad, en los datos que se manejan en el proyecto SGF para una posible migración hacia una versión inferior de PostgreSQL de la que están utilizando actualmente se encuentran en:

- El número de argumentos de una función.

En cuanto a una posible migración a Oracle los mayores problemas de portabilidad en sus datos se encuentran en:

- Uso de los tipos de datos Booleano, Char y Timestamp with time zone.
- Generación de claves automáticas.
- Uso de valores NULL en la expresión Order By.

En el proyecto Prisiones donde el SGBD utilizado es Oracle la herramienta arrojó como resultado un 34% de afectación y un 66% de portabilidad (Fig. 3.3) para una posible migración hacia PostgreSQL.

| | A | B | C | D |
|---|---|---|---|---|
| 1 | ***Métricas*** | ***Valor de uso*** | ***Costo*** | ***Valor de Portabilidad*** |
| 2 | Uso de actualizaciones de vistas. | 0,2 | 1 | 0,2 |
| 3 | Uso de la cláusula LIKE | 1 | 1 | 1 |
| 4 | Utilización del límite simple | 0,5 | 0,6 | 0,3 |
| 5 | Inserción múltiple de tuplas | 0,4 | 1 | 0,4 |
| 6 | Uso del tipo de datos BOOLEANO | 0,1 | 0,4 | 0,04 |
| 7 | Uso del tipo de datos CHAR | 0 | 0,5 | 0 |
| 8 | Uso del tipo de datos TIMESTAMP WITH TIME ZONE | 0,2 | 0,4 | 0,08 |
| 9 | Uso de la función CHARACTER_LENGTH | 0,4 | 1 | 0,4 |
| 10 | Uso de la función SUBSTRING | 0,3 | 1 | 0,3 |
| 11 | Concatenación de columnas que admitan valores nulos. | 0,2 | 0,5 | 0,1 |
| 12 | Obtención de descripciones de consultas | 0,1 | 1 | 0,1 |
| 13 | Obtención del tiempo de consulta | 0,1 | 1 | 0,1 |
| 14 | Generación de claves automáticas | 1 | 1 | 1 |
| 15 | Utilización de valores NULL en la expresión Order By. | 0,4 | 1 | 0,4 |
| 16 | | | | |
| 17 | ***% de afectación de la portabilidad:*** | | | ***34,00*** |
| 18 | | | | |
| 19 | ***% de portabilidad del modelo analizado:*** | | | ***66,00*** |
| 20 | | | | |
| 21 | ***Escala del valor de uso*** | | | |
| 22 | Si la característica no se usa el valor es: 0 | | | |
| 23 | Si la característica se usa poco el valor puede estar entre 0,1 y 0,4 | | | |
| 24 | Si la característica se usa en gran medida el valor puede estar entre 0,5 y 1 | | | |

Ready

Fig. 3.3 Resultados de la validación en el proyecto SGP.

En el proyecto SGP la portabilidad para una posible migración desde Oracle hacia PostgreSQL se ve afectada en todas las métricas establecidas para evaluar la portabilidad con la excepción del uso de Tipos de datos char.

En el proyecto SAREN igualmente el SGBD utilizado es Oracle y los resultados obtenidos luego de evaluar la información que en ese se maneja a través de la herramienta son de un 61.5% de afectación de la portabilidad siendo su base de datos solo un 38.5% portable al SGBD PostgreSQL ( Fig.3.4).

| | A | B | C | D |
|---|---|---|---|---|
| 1 | ***Métricas*** | ***Valor de uso*** | ***Costo*** | ***Valor de Portabilidad*** |
| 2 | Uso de actualizaciones de vistas. | 0 | 1 | 0 |
| 3 | Uso de la cláusula LIKE | 0 | 1 | 0 |
| 4 | Utilización del límite simple | 1 | 0,6 | 0,6 |
| 5 | Inserción múltiple de tuplas | 1 | 1 | 1 |
| 6 | Uso del tipo de datos BOOLEANO | 0,5 | 0,4 | 0,2 |
| 7 | Uso del tipo de datos CHAR | 0,5 | 0,5 | 0,25 |
| 8 | Uso del tipo de datos TIMESTAMP WITH TIME ZONE | 0 | 0,4 | 0 |
| 9 | Uso de la función CHARACTER_LENGTH | 0 | 1 | 0 |
| 10 | Uso de la función SUBSTRING | 0,5 | 1 | 0,5 |
| 11 | Concatenación de columnas que admitan valores nulos. | 1 | 0,5 | 0,5 |
| 12 | Obtención de descripciones de consultas | 1 | 1 | 1 |
| 13 | Obtención del tiempo de consulta | 1 | 1 | 1 |
| 14 | Generación de claves automáticas | 1 | 1 | 1 |
| 15 | Utilización de valores NULL en la expresión Order By. | 0,1 | 1 | 0,1 |
| 16 | | | | |
| 17 | ***% de afectación de la portabilidad:*** | | | ***61,50*** |
| 18 | | | | |
| 19 | ***% de portabilidad del modelo analizado:*** | | | ***38,50*** |
| 20 | | | | |
| 21 | ***Escala del valor de uso*** | | | |
| 22 | Si la característica no se usa el valor es: 0 | | | |
| 23 | Si la característica se usa poco el valor puede estar entre 0,1 y 0,4 | | | |
| 24 | Si la característica se usa en gran medida el valor puede estar entre 0,5 y 1 | | | |

Ready

Fig. 3.4 Resultados de la validación en el proyecto SAREN.

Los resultados obtenidos muestran las posibilidades que existen en cada caso de llevar a cabo una migración ya sea a una versión anterior a la que estén utilizando en el caso de los que usan como SGBD PostgreSQL o para hacer una migración de un SGBD a otro para los casos de PostgreSQL y Oracle.

Con el uso de esta herramienta se reduce en gran medida el tiempo que muchas veces se invierte en una migración sin saber si al final dará resultados o no, pues haciendo uso de esta se puede evaluar cuan portable es la información que se maneja y si es posible llevar a cabo esa migración.

En el proyecto SAREN, donde el porciento de afectación de la portabilidad es elevado, la mayor afectación para una posible migración desde Oracle hacia PostgreSQL se encuentran en:

- Utilización del límite simple
- Inserción múltiple de tuplas
- Uso del tipo de datos BOOLEANO
- Uso del tipo de datos CHAR
- Uso del tipo de datos TIMESTAMP WITH TIME ZONE

- Uso de la función CHARACTER_LENGTH
- Uso de la función SUBSTRING
- Concatenación de columnas que admitan valores nulos.
- Obtención de descripciones de consultas
- Obtención del tiempo de consulta
- Generación de claves automáticas
- Utilización de valores NULL en la expresión Order By.

## 3.3. Conclusiones

Con la realización de la Herramienta para evaluar portabilidad de los datos se le da cumplimiento al objetivo de esta investigación.

Para comprobar que la misma surte el efecto deseado se aplicó en 3 de los proyectos de la UCI, arrojando como resultado que los datos que se manejan en estos no son portables. Esta validación permitió llegar a las conclusiones de que los proyectos con mayor afectación son los que utilizan como SGBD Oracle y que además son proyectos internacionales, puesto que en ellos se explotan al máximo las facilidades que brinda el gestor.

# 4. Conclusiones generales

Teniendo en cuenta la importancia que tiene el almacenamiento de información en el desarrollo de software y para poder cumplir con el objetivo trazado:

- De acuerdo a las características de los SGBD más utilizados hoy en día, se pudo identificar: a partir de los métodos investigativos como la encuesta; y la heterogeneidad de los proyectos que se realizan en la UCI, que los SGBD seleccionados para realizar la herramienta son PostgreSQL y Oracle.
- A partir de las características de los gestores seleccionados, las cuales identifican sus modelos de datos, consultas y operaciones se establecieron métricas para evaluar la portabilidad de los datos.
- Con las métricas establecidas se desarrolló la herramienta con el objetivo de medir el grado de portabilidad de los modelos de datos, de aquellos proyectos que emplean los gestores PostgreSQL y Oracle en la UCI.

Luego de validar la herramienta se pudo concluir que los mayores problemas de portabilidad se encuentran en los proyectos que utilizan Oracle y que son proyectos internacionales, cuyas deficiencias se encuentran fundamentalmente en:

- La inserción múltiple de tuplas.
- Uso de la función SUBSTRING.
- Generación de claves automáticas.

Luego de todo el análisis se puede decir que la herramienta cumple con el objetivo por el cual fue elaborada, puesto que se comprobó que la misma es capaz de detectar los problemas de portabilidad de los datos que puedan existir en determinado proyecto.

# 5. Recomendaciones

Se recomienda:

- Extender la herramienta para medir la portabilidad entre versiones de Oracle como mismo se hizo con las versiones de PostgreSQL.
- Añadirle a la herramienta otros SGBD utilizados en el desarrollo de software en la Universidad de las Ciencias Informáticas.
- El perfeccionamiento y posible elaboración de la herramienta de una forma más robusta.
- Incrementar algunas características que también deben ser tenidas en cuenta para evaluar portabilidad de la información, entre las cuales se pueden mencionar: el uso de tablespaces y el límite por subconsultas.
- Enriquecer la investigación según vayan surgiendo nuevas versiones de PostgreSQL y Oracle.
- Crear una línea base con los resultados provistos por la herramienta con el fin de refinar los pesos establecidos para cada métrica, para lograr indicadores de portabilidad cada vez más realistas.

## 6. Bibliografía

1. **Garcia, Daysi Karina.** *Gestión de la Calidad, Guía Didáctica.* Loja : s.n., 2008.

2. **Infocalidad.** Infocalidad.net. [En línea] 2005. [Citado el: 15 de Diciembre de 2009.] http://www.infocalidad.net/gest_calidad_def/definicion.asp.

3. Calidad. [En línea] http://calidad.umh.es/curso/criterio.htm.

4. **Quiñones, Ernesto.** *Modelos de Calidad de Software y Software Libre.*

5. Certificación de Normas. [En línea] http://www.deltellyasociados.com/certificacion.php.

6. **Pressman.** *Ingeniería del Software, Un enfoque práctico.* 2006.

7. **Westfall.** *"Software metrics that meet your information needs.".* 1995.

8. **IEEE.** *"Software Engineering Standards Std. 6 10.12-1990." .* 1993.

9. **Bryant, J. Cartty.** *Control de Calidad.* México : Pax, 1998.

10. **Mooney.** *Traer portabilidad al proceso del software.* 1997.

11. **Allen B. Tucker, Robert Noonan.** *Paradigmas de los lenguajes de programación.* España : s.n.

12. **González, Luis Alberto Pimentel.** "ArBaWeb: Arquitectura base sobre la Web". 2007.

13. **Suárez, Mainoldis Fuentes.** Propuesta de arquitectura para sistemas de gestión hospitalaria. 2007.

14. **Ecured.** Ecured. [En línea] http://www.ecured.cu/index.php/Mysql.

15. **Garcia, Eudy.** *Propuesta de Entorno de Desarrollo Para el Centro de Automatización de la Dirección y la Información Basada en el Uso de Software Libre. .* 2007.

16. **Equinoccial, Universidad Tecnológica.** *Libro Excel Avanzado.*

17. **Sicilia, Miguel-Angel.** *Estándar ISO 9126 del IEEE y la Mantenibilidad.* 2009.

18. **Sanchez, Luis Aguila.** *Control de la Calidad.* s.l. : Minerva, 1997.

19. **S, Kan.** *Metrics and Models in Software Quality Engineering.* 2002.

20. **Quiñones, Ernesto.** Apesol. [En línea] 2009. http://www.apesol.org..

21. **PostgreSQL.** postgresql.org. [En línea] 2009. [Citado el: 12 de Enero de 2010.] http://www.postgresql.org/docs/8.4/static/sql-commands.html.

22. —. postgresql.org. [En línea] http://www.postgresql.org/about/featurematrix.

23. **Postgresql.** Postgresql. [En línea] http://www.postgresql.org.

24. **Pecos, Daniel.** Danielpecos. [En línea] 2009. http://danielpecos.com/docs/mysql_postgres/x15.html.

25. **Oracle.** Oracle. [En línea] 2009. http://download.oracle.com/docs/cd/E11882_01/server.112/e10592/toc.htm.

26. —. Oracle. [En línea] http://oracle.com.

27. **MSDN.** Microsoft. [En línea] [Citado el: 13 de Eenero de 2010.] http://msdn.microsoft.com/es-es/library/ms177420.aspx.

28. **Microsoft.** Microsoft. [En línea] http://msdn.microsoft.com/es-es/library/ms177420.aspx.

29. —. Microsoft. [En línea] http://support.microsoft.com/kb/211485/es.

30. **merick, Jerry.** *Administrando el Almacenamiento de Datos XML.*

31. **L.Briand, J. Daly, C.Differding.** *An experimental comparison of the maintainability of objectoriented.* 1996.

32. **Java.** Java.sun.com. [En línea] http://java.sun.com/docs/books/tutorial/jdbc/basics/index.html.

33. **ISO.** *iso8601.*

34. —. ISO. [En línea] http://www.iso.org/iso/home.htm.

35. **ISBN.** *Diccionario de Informática, Portabilidad.* Madrid, España. : Cultural, 1999.

36. **Gómez, José Joaquín Mira y José Mª.** *Criterio, Indicador y Estándar.*

37. **Giraldo, Otoniel Perez.** *Métricas, Estimación y Planificación en Proyectos de Software.* Guadalajara. : s.n.

38. **Gevara, Yusmilia.** *Sistema de Manejo de Datos de Ensayos Clínicos: Diseño de la Arquitectura. .*

39. **Garey.** *Portabilidad del software: Pesando las opciones, haciendo opciones. .* 2007.

40. **Fajardo, Norge.** *Sistema de réplica para bases de datos distribuidas en PostgreSQL.* 2007.

41. **Castellano, Maria.** *Calidad tota.* México : La prensa Medica, 1998.

42. **Bamnet, Jeanne.** *Control de la Calidad.* Barcelona España : Fontanella, 1991.

43. **Argentina, PostgreSQL.** Portal de la comunidad de PostgreSQL Argentina . [En línea] http://www.arpug.com.ar/trac/wiki/sql-vacuum.html.

44. **Anastasi, Maribel.** *Control de Calidad.* Lima : AGUILAR, 1992.

45. **Acm.** Acm. [En línea] http://www.acm.org/crossroads/espanol/xrds8-4/XML_RDBMS.html.

46. Unerg. [En línea] http://www.unerg.edu.ve/index.php?option=com_docman&task=doc_view&gid=160.

47. Troels.arvin.dk. [En línea] [Citado el: 20 de Febrero de 2010.] http://www.troels.arvin.dk/db/rdbms/.

48. Pgfoundry. [En línea] 2009. http://pgfoundry.org/..

49. Kioskea. [En línea] http://es.kioskea.net/contents/qualite/qualite-introduction.php3.

# 7. Anexos

## 1- Encuesta realizada al inicio de la investigación.

Nombre del proyecto:_____

1. ¿Qué gestor de base de datos utiliza en su proyecto? Diga que versión del mismo emplean.

   __ Oracle. __ SQL Server. __ Postgre SQL

   __ My SQL. __ Otro.

2. ¿Para las consultas a la BD usa SQL o algún otro lenguaje particular del SGB?

   __ SQL. __ Otro.

a) ¿En caso de ser otro, diga cuál?

3. ¿Han pensado en la posibilidad de una futura migración hacia otras versiones del mismo gestor que usan en el proyecto o a otro gestor?

   __ Sí. __ No.

   a) ¿En caso de ser positivo como tienen pensado hacerlo?

4. ¿Conocen o utilizan algún mecanismo para evaluar la portabilidad de los datos?

   __ Sí. __ No.

   a) En caso de ser positivo diga cuál.

5. ¿En qué codificación se encuentran los datos de su BD?

## 2- Cartas de validación de métricas establecidas en el trabajo.

Validación de métricas para evaluar portabilidad de la información en las bases de datos.

En el Trabajo de Diploma para optar por el título de Ingenieros en Ciencias Informáticas, que tiene como tema Herramienta para evaluar la portabilidad de los datos en el desarrollo de software, de los autores Alicia del Carmen Pérez Roldán y Jorge Luis Fuoman Noguera, se establecen métricas para evaluar la información que se maneja en el desarrollo de un determinado software. Se hace necesario hacer una validación de estas métricas para avalar que el resultado de las mismas será totalmente satisfactorio por lo que se le solicita que dé su opinión acerca de estas.

Opina: Ing. Slalymar Avilés Carrazana

Proyecto: RNA

Cargo: DBA

Años de experiencia: 5

Opinión:

Las métricas analizadas en el trabajo de diploma satisfacen la necesidad que se plantea a la hora de evaluar la portabilidad de los datos en el desarrollo de software, considero que las analizadas en el Trabajo ayudan a entender mejor y a poder portar los datos de un gestor a otro con una mayor facilidad ya que se contaría de antemano con un estudio sobre el tema, no obstante considero que quedan elementos que son importantes que deben ser estudiados con profundidad en una posible continuidad del trabajo.

Firma

Validación de métricas para evaluar portabilidad de la información en las bases de datos.

En el Trabajo de Diploma para optar por el título de Ingenieros en Ciencias Informáticas, que tiene como tema Herramienta para evaluar la portabilidad de los datos en el desarrollo de software, de los autores Alicia del Carmen Pérez Roldán y Jorge Luis Fuoman Noguera, se establecen métricas para evaluar la información que se maneja en el desarrollo de un determinado software. Se hace necesario hacer una validación de estas métricas para avalar que el resultado de las mismas será totalmente satisfactorio por lo que se le solicita que dé su opinión acerca de estas.

Opina: Marcos Ortiz Valmaseda

Proyecto: Jefe del Grupo "Personalización de Postgres SQL"

Cargo: _____

Años de experiencia: 4 años

Opinión: las métricas definidas en la presente investigación pueden usarse correctamente para evaluar la portabilidad, aunque hay algunas que no están presentes en el documento. Algunas de éstas son:

- Uso de tablespaces (costo 9)
- Limit por subconsulta (costo 0.6)

Mi opinión es que use la herramienta que defina dichas métricas en la mayoría de los proyectos que se usan en la Universidad sobre Oracle, para poder tener una idea bastante clara y concisa de cuán complejo resultaría la migración hacia Postgres SQL.

Firma

Validación de métricas para evaluar portabilidad de la información en las bases de datos.

En el Trabajo de Diploma para optar por el título de Ingenieros en Ciencias Informáticas, que tiene como tema Herramienta para evaluar la portabilidad de los datos en el desarrollo de software, de los autores Alicia del Carmen Pérez Roldán y Jorge Luis Fuoman Noguera, se establecen métricas para evaluar la información que se maneja en el desarrollo de un determinado software. Se hace necesario hacer una validación de estas métricas para avalar que el resultado de las mismas será totalmente satisfactorio por lo que se le solicita que dé su opinión acerca de estas.

Opina: José Alejandro Lugo García

Proyecto: CALISOFT

Cargo: Especialista

Años de experiencia: 4

Opinión:

Luego de haber realizado un estudio de la propuesta, considero que el propósito de la herramienta tiene validez y posee utilidad para ser empleada en entornos reales de producción. Los autores presentan nuevas métricas de portabilidad que emplean factores de peso subjetivos. Es necesario crear una línea base con los resultados provistos por la herramienta con el fin de refinar dichos pesos para lograr indicadores de portabilidad cada vez más realistas.

Como recomendación, los autores deberán tener en cuenta aquellas métricas de portabilidad planteadas por la NC-ISO 9126 y adicionarlas en futuras versiones de la herramienta resultado de la investigación.

Firma

## 3- Acta de evaluación del trabajo emitida por el proyecto Automatización y Modernización de la Coordinación de Antecedentes Penales de Venezuela.

Los profesores del proyecto **Automatización y Modernización de la Coordinación de Antecedentes Penales de Venezuela**, después de analizar el contenido de la tesis de grado **Herramienta para evaluar la portabilidad de los datos en el desarrollo de software**, de los autores Jorge Luis Fuoman Noguera y Alicia del Carmen Pérez Roldán, esta última integrante del proyecto, desarrollaron las siguientes observaciones:

La herramienta desarrollada es aplicable directamente a los dos sub-proyectos, ya que permitirá evaluar la portabilidad de los datos de los modelos propuestos para los sistemas y de la base de datos del sistema que se utiliza actualmente con vistas a facilitar la migración. Estas evaluaciones servirán de apoyo al diseño de modelos de datos más eficientes para la ejecución del sistema que se desarrollará. El trabajo tiene buena calidad tanto del contenido como de su organización y su generalización puede contribuir al elevar la calidad de las bases de datos de los proyectos.

Proyecto de Automatización y Modernización de la Coordinación de Antecedentes Penales de Venezuela, mayo 18 de 2010.

Ing. Juniel Tamayo Hernández
Líder de proyecto (en funciones)

# 8. Glosario de Términos

1. **ANSI:** Instituto Nacional Estadounidense de Estándares (ANSI, por sus siglas en inglés: American National Standards Institute) es una organización que supervisa el desarrollo de estándares para productos, servicios, procesos y sistemas en los Estados Unidos.
2. **Internet Engineering Task Force (IETF):** (en español Grupo de Trabajo en Ingeniería de Internet) es una organización internacional abierta de normalización, que tiene como objetivos contribuir la ingeniería de Internet, actuando en diversas áreas, como transporte, encaminamiento, seguridad. Fue creada en EE. UU. en 1986. La IETF es mundialmente conocida por ser la entidad que regula las propuestas y los estándares de Internet, conocidos como RFC.
3. **GNU GPL:** La Licencia Pública General de GNU (en inglés GNU General Public License) o simplemente su acrónimo del inglés GNU GPL, es una licencia creada por la Free Software Foundation en 1989, y está orientada principalmente a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.
4. **Vista:** Una vista en base de datos es un resultado de una consulta SQL de una o varias tablas.
5. **Disparador:** Un disparador también conocido como **Trigger** en una Base de datos, es un procedimiento que se ejecuta cuando se cumple una condición establecida al realizar una operación de inserción (INSERT), actualización ( UPDATE) o borrado (DELETE).
6. **Ficheros**: Agrupación de información que puede ser manipulada por el sistema operativo de un ordenador. Un fichero puede tener cualquier tipo de contenido: texto, ejecutables, gráficos, etc.
7. **IEEE:** Corresponde a las siglas de Instituto de Ingenieros Electricistas y Electrónicos, una asociación técnico-profesional mundial dedicada a la estandarización.

8. **Software**: Componentes intangibles de una computadora, conjunto de programas y procedimientos necesarios para hacer posible la realización de una tarea específica.
9. **Plataforma**: Es el principio sobre el cual un software puede ejecutarse o desarrollarse.
10. **Procedimiento almacenado**: Un procedimiento almacenado (*stored procedure* en inglés) es un programa (o procedimiento) el cual es almacenado físicamente en una base de datos. Posee acceso directo a los datos que se encuentran en la base de datos.
11. **Consultas:** Se definen preguntas a la base de datos con el fin de extraer y presentar información contenida en las mismas.
12. **XQuery:** Lenguaje de consulta XML publicado por el W3C (World Wide Web Consortium) para acceso concurrente a datos ordinarios SQL y documentos XML.
13. **TRUNCATE**: Este comando trunca todo el contenido de una tabla. La ventaja sobre el comando DELETE, es que si se quiere borrar todo el contenido de la tabla, es mucho más rápido, especialmente si la tabla es muy grande, la desventaja es que TRUNCATE solo sirve cuando se quiere eliminar absolutamente todos los registros, ya que no se permite la cláusula WHERE
14. **Web**: Nombre con que se nombra a la World Wide Web. Sistema de comunicación y de publicación que fue diseñado para distribuir información a través de redes de computadoras.
15. **Microsoft**: Casa desarrolladora de software, creadora de sistemas operativos, así como de aplicaciones informáticas de todo tipo.
16. **Sybase**: Compañía dedicada al desarrollo de tecnología de la información.
17. **Mainframes**: Computadora de gran tamaño de tipo multiusuario, utilizada en empresas.
18. **Data warehousing**: En el contexto de la informática, un **almacén de datos** (del inglés *data warehouse*) es una colección de datos orientada a un determinado ámbito ( empresa, organización, etc.)

19. **Clúster**: Un clúster (o unidad de asignación según la terminología de Microsoft) es un conjunto contiguo de sectores que componen la unidad más pequeña de almacenamiento de un disco.
20. **Commit:** La instrucción COMMIT garantiza que todas las modificaciones de la transacción se conviertan en una parte permanente de la base de datos. La instrucción COMMIT también libera recursos que utiliza la transacción como, por ejemplo, los bloqueos.
21. **VAX-VMS**: Sistema Operativo propietario de Digital Equipment Corporation (DEC) para sus máquinas VAX.
22. **IBM:** International Business Machines o IBM (por sus siglas en inglés) es una empresa multinacional que fabrica y comercializa herramientas, programas y servicios relacionados con la informática.
23. **SQL Anywhere**: Refiérase a tecnología de Bases de Datos optimizadas para usar fuera de un centro de datos tradicional.
24. **ISO:** Organización Internacional para la Estandarización. Su función principal es la de buscar la estandarización de normas de productos y seguridad para las empresas u organizaciones a nivel internacional.
25. **XML:** Lenguaje que presenta un conjunto de reglas que sirven para definir etiquetas semánticas para organizar documentos de la Web.
26. **API Flexible:** Interfaz de programación de aplicaciones o API (del inglés application programming interface) es el conjunto de funciones y procedimientos que ofrecen cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Usados generalmente en las bibliotecas.